ARMY RESEARCH LABORATORY

Design of a Heart Sound Extraction Algorithm for an Acoustic-Based Health Monitoring System

by Steven R. Murrill and Michael V. Scanlon

ARL-MR-517 October 2002

Approved for public release; distribution unlimited.

20030109 133

The findings in this report are not to be construed as an official Department of the Army position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Army Research Laboratory
Adelphi, MD 20783-1197

ARL-MR-517 **October 2002**

Design of a Heart Sound Extraction Algorithm for an Acoustic-Based Health Monitoring System

Steven R. Murrill and Michael V. Scanlon
Sensors and Electron Devices Directorate, ARL

Approved for public release; distribution unlimited.

Acknowledgments

The authors would like to thank Dr. Russell McCally and Dr. Larry Younkins of the Johns Hopkins Applied Physics Laboratory for approving and academically supporting a substantial portion of this work as part of a Johns Hopkins Special Projects 800-level graduate course. The authors also thank Dr. Larry Younkins for his technical support, mentoring, and guidance of Steven Murrill both during the Digital Signal Processing part of the course and throughout the duration of the follow-on project-execution phase of the course.

Steven Murrill would also like to thank the co-author, Michael Scanlon of the U.S. Army Research Laboratory, for his assistance in defining a meaningful and relevant project, for his technical suggestions and guidance, for his invaluable help in orchestrating and participating in the data collection activity, and for his contributions to the cross-correlation algorithm development and analysis activities.

The authors would also like to thank David Gonski for providing the continuous-time, anti-aliasing prefilter circuitry that was an integral part of the PC-based acquisition system used to capture the ECG and acoustic data utilized in this work.

Contents

Figures

Tables

1. Introduction

The U.S. Army Research Laboratory (ARL) has developed a new method to measure human physiological stress parameters. This consists of an acoustic sensor positioned inside a fluid-filled bladder in contact with the human body. Packaging the sensor in this manner minimizes outside environmental interferences, while signals within the body are transmitted to the bladder with minimal losses. This fluid-coupling technology comfortably conforms to the human body and enhances the signal-to-noise-ratio (SNR) of human physiology to that of ambient noise. This sensor is not readily available because its development has not yet been completed. An acoustic sensor of this type could be a tremendous asset in determining soldier stress levels during demanding tasks.

An acoustic sensor system can detect changes in a person's physiological status resulting from exertion or trauma such as penetrating wounds, hypothermia, dehydration, heat stress, and many other medical conditions (or illnesses). Indications of a dangerous condition can be used to recommend corrective procedures or to simply alert medical personnel or supervisors. Acoustic sensors and signal processing may allow the prediction of injury or unsafe actions, based on advance knowledge of health and performance trends gathered during the interactions between a soldier and his mission or between a worker's performance on the job and the man/machine interface in the workplace. Managers can use preparedness and physiological data as a decisional aid for human resource allocations. Training leaders and participants can monitor performance levels for the presence of dangerous physiological conditions. Additionally, the data collected during training or routine tasks can be used in predictive modeling and simulation of worker performance in a virtual workplace, especially in development of new operative environments or procedures. Other civilian technology transfer applications include sudden infant death syndrome (SIDS), apnea, and infant monitoring as well as clinical surveillance in convalescent and Veterans Administration (VA) homes, medical transports, hospitals, and telemedicine applications [1–4]. Drivers of vehicles and aircraft could also be monitored for the onset of sleep, seizure, or heart attack.

From a diagnostic perspective, changes in the interval between heartbeats, known as the interbeat interval (IBI), are of physiological significance. These changes in IBI, otherwise known as heart rate variability (HRV), are a measure of mental workload; HRV typically decreases as effort invested in a task increases. Fast Fourier transform (FFT) analysis of HRV can be divided into three different control regimes: low-, medium-, and high-frequency peaks (or bands) relate to body temperature regulation, short-term arterial pressure regulation, and respiratory activity, respectively [5]. Changes in valve timing may also provide clues on cardiac function and overall physiology. Heart sound IBI measurements can be taken at the wrist, neck, temple, or chest. However, valve sounds are available primarily at the chest area with some components discernable in the neck region.

Fourier analysis of the monitoring pad's output has already shown that human cardiopulmonary function contains infrasonic (sounds below 20 Hz) signals that cannot be heard by the human ear, but may be useful for physiological monitoring and medical diagnostics. Spectral details of individual valve and chamber activity can be monitored for timing and qualitative factors as

well. For example, the "first heart sound" is a result of the mitral and tricuspid valves closing, whereas the "second heart sound" results from the aortic and pulmonary valves closing. When inhaling, the interval between the aortic and pulmonary valve closures increases, allowing the two components of the "second heart sound" to be heard separately. By monitoring the amplitudes of the first heart sounds, which are correlated to the left ventricle pressures, cardiac contractility can be measured [6]. Systolic blood pressure values for an individual patient can be approximated from sound-pattern analysis of the second heart sound and can be considered a qualitative measure. The correlation of these values with a known systolic measurement adds a quantitative baseline that provides greater precision [7].

2. Purpose

The purpose of this project was to develop a rudimentary, first and second heart sound IBI extraction algorithm for the experimental, acoustic-based health monitoring system previously described, using only basic concepts and techniques generally known/available to first-semester Discrete-Time Signal Processing course graduates. The primary goal and challenging/critical aspect of this work was to develop a heart sound extraction algorithm that is capable of extracting *instantaneous* IBIs/heart rates on a beat-by-beat basis rather than to develop an algorithm that is simply capable of extracting *averaged* IBIs/heart rates.

Figure 1 illustrates the two basic, *instantaneous* IBIs (or *instantaneous* heart rates) of interest, namely the IBI between the first heart sound of one heart beat and the first heart sound of the next heart beat (IBI 1-1*) and the IBI between the second heart sound of one heart beat and the second heart sound of the next heart beat (IBI 2-2*), along with all of the subintervals between the first and second heart sounds of two consecutive heart beats.

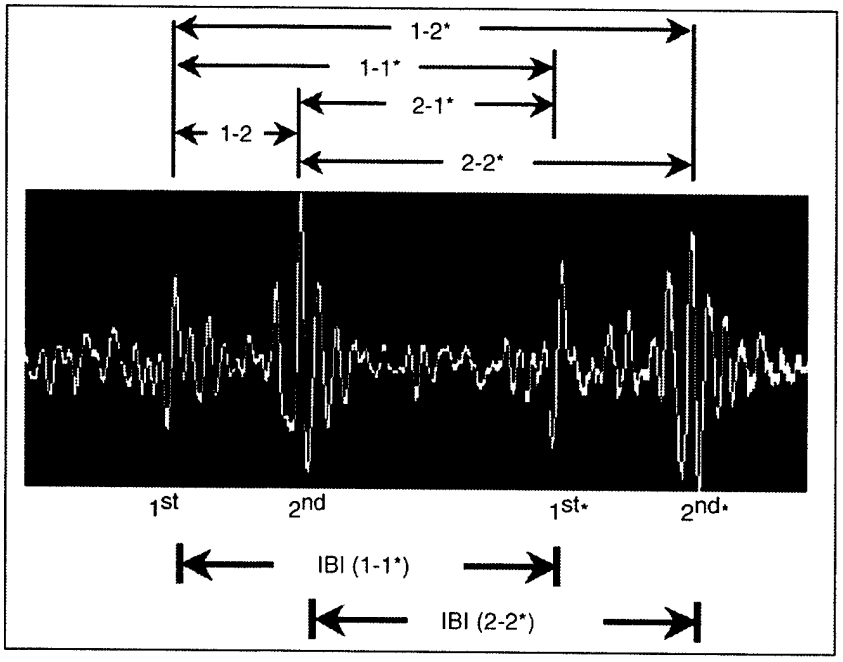

Figure 1. Heart sound IBI and subinterval definitions.

3. Design Approach

In consideration of the motivation described previously and with particular regard to the mobile application space of the envisioned acoustic-based physiological monitoring systems, two primary (and generally competing) objectives arise: to maximize algorithm robustness to the varied nature of the operational environment, e.g., background interferences, and to minimize the power consumption of the hardware implementing the algorithm.

The approach taken for this project was to place higher priority on providing a nominal degree of background interference rejection, while giving some consideration to computational efficiency.

As a means of validating the acoustic-sensor-based heart sound extraction algorithm, an experiment was devised in which both acoustic-sensor and electrocardiogram (ECG) data was captured simultaneously using two different subjects during separate trials. The experiment is described in section 6.

The systematic approach used to develop the discrete-time signal-processing algorithm was to perform a spectral analysis of typical acoustic and ECG signals acquired in both a quiet (benign) and a high-noise (background) environment, develop optimal discrete-time filtering processes, and develop an ECG and IBI (first and second) heart sound extraction process.

For this project, the acoustic (heart sound containing) signals to be extracted were taken from a gel-coupled sensor positioned against the neck area of the test subject (see Figure 2). Hardware for mounting these sensors, such as straps and chest harnesses, has been developed at ARL. Mounting mechanisms have been designed with the flexibility necessary to allow sensors to be mounted in several different locations on the human body. However, it is recognized that an optimal configuration for one body location may not work as well on other areas, and there are significant tradeoffs to be considered for placement of sensors at different body locations. One of these tradeoffs is user acceptance. If the user (test subject) does not like the attachment location, sensor placement, or attachment method because it is uncomfortable or because the sensor system interferes with his normal activity or abilities, it will adversely affect the test/mission and will not be useful. The neck placement area is of considerable interest because it has the potential to provide a good combination of user acceptability, strong breath, and voice sounds along with

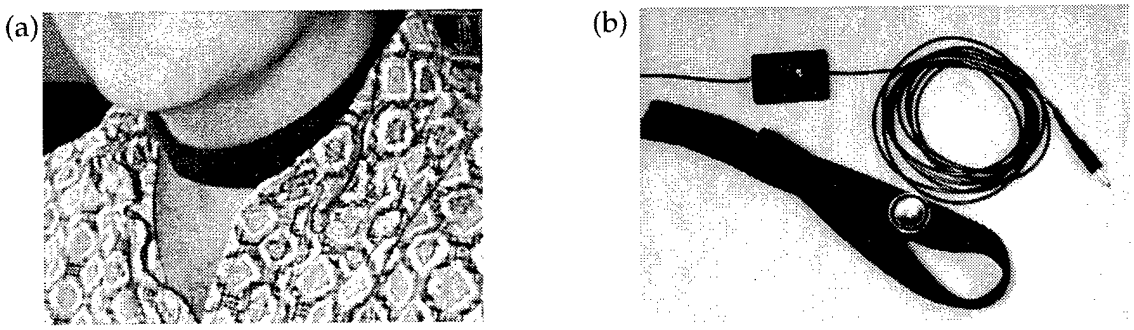

Figure 2. Neck-placed acoustic sensor: (a) typical neck placement and (b) sensor mounting hardware.

useable heart sound levels. The algorithm design implication(s) of this sensor placement are straightforward; a high degree of breath and voice sound rejection will be required to reliably extract IBI heart sound information.

Another design consideration is the nature of the acoustic sensor itself. The experimentally developed acoustic sensor used in this research was configured with a thin flexural-disk piezoelectric element within a fluid chamber. Preliminary analysis indicated that this configuration would provide a useful response bandwidth up to 2500 Hz. Experimentation led to the selection of this thin, flexural-disk, piezoceramic element that improved sensor bandwidth and sensitivity over earlier devices. Additionally, the exposed surface of the new low-cost sensing element is resistant to corrosion or failures due to continuous submersion in liquid. Other sensor materials such as piezoelectric rubber (PZR), 1,3 piezocomposites, lead zirconate titanate (PZT), electret, etc., were gathered and evaluated, but were considered inappropriate for the implementation or did not meet sensitivity and bandwidth goals. The piezoelectric material deposited on the flexible metal membrane is somewhat brittle, and care must be taken not to push directly on the sensor face since microcracks resulting from overstressing the diaphragm may cause decreased sensitivity. Other materials such as polyvinylidene fluoride (PVDF), a flexible piezoelectric material, are more durable and conform better to the contours of the human body. This and other materials and sensors are still being evaluated. A sensor prototype, consisting of a piezoelectric disk, housing, and fluid cavity, was designed and fabricated and is shown in Figure 3. Sensor assembly and cross-sectional drawings are shown in Figures 4 and 5.

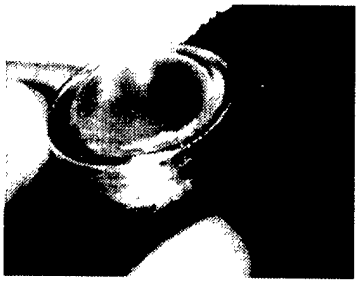

Figure 3. Gel-coupled substrate.

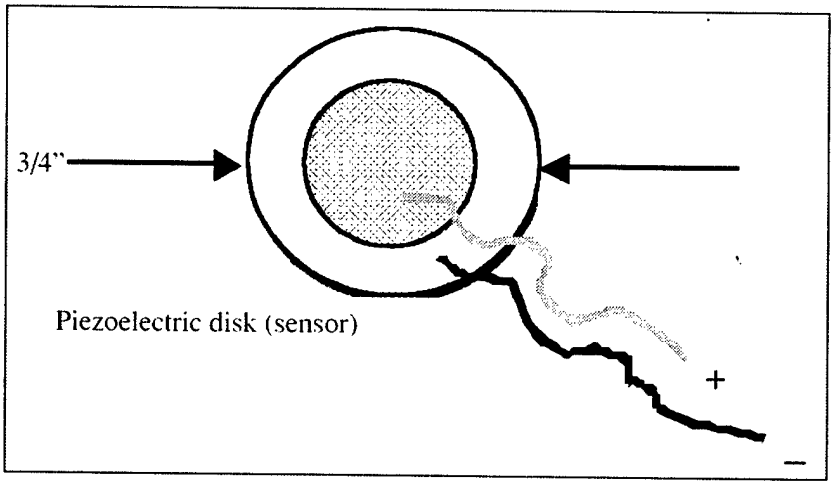

Figure 4. Piezoelectric disk mounted on brass acoustic sensor.

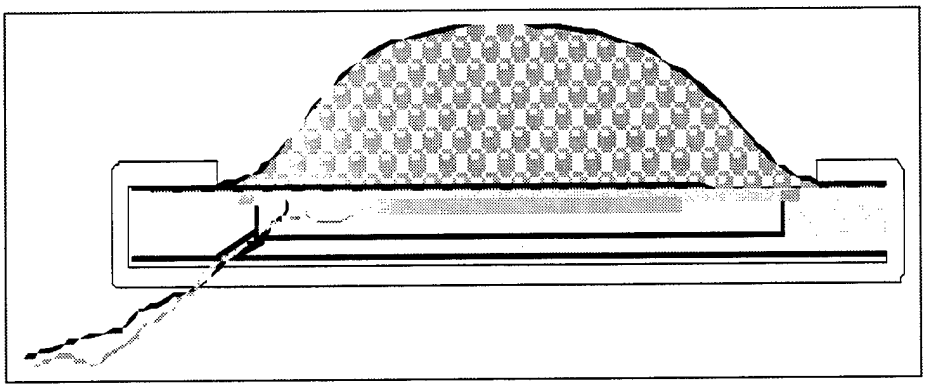

Figure 5. Sensor cross section.

The general design assumptions and constraints used in the development of both the acoustic and the ECG IBI signal extraction algorithm were: the algorithm is only required to extract IBIs associated with heart beat rates between 20 and 240 beats/min, and the primary emphasis is to provide a high degree of timing accuracy along with a nominal degree of background interference rejection, while giving some consideration to computational efficiency.

Additional constraints and criteria that were applied to the acoustic IBI signal extraction algorithm are detailed in section 5.

4. Spectral Analysis

The algorithm development process began with a spectral analysis of representative samples of both the acoustic and the ECG signals. The approach used for the spectral analysis was to perform multiple chirp z-transform calculations [8] across different frequency ranges from 0 to 3 Hz all the way up to 0–750 Hz on both acoustic and ECG signal samples. In order to achieve good *spectral* resolution, several chirp z-transforms/spectrograms were performed on signal sequences using window lengths that spanned several cycles of heart beat/heart sound activity (up to 16384 points). In order to obtain good *time* resolution, chirp z-transforms were performed on signal sequences using window lengths that progressively spanned smaller and smaller subcycle intervals of heart beat/heart sound activity (typically from 256 points down to 32 points). The Hamming window was chosen as the window shape for this analysis because it offers a good combination of peak side-lobe rejection (approximately –40 dB) and good spectral resolution (main lobe width approximately 8 /window length [in samples]) [8]. All signals were acquired with a PC-based acquisition system that utilized 12-bit analog-to-digital converters (A/Ds) set to sample at a rate of 1500 Hz. To minimize quantization error, the dynamic range of the A/D acquisition card was limited to ±1.0 V. To avoid aliasing, the acoustic and ECG continuous-time signals were first pre-filtered using a Maxim MAX293 8th-Order, Low-Pass, Elliptic, Switched-Capacitor Filter* circuit where the corner frequency was set for 500 Hz. The actual corner frequency was measured at approximately 520 Hz.

Although several acoustic signal sequences were available, the spectral analysis of the acoustic sensor data was primarily focused on a captured sequence where, for the first 15 s, the test

*Maxim Integrated Products. 120 San Gabriel Drive, Sunnyvale, CA 94086.

subject was holding his breath, at the 15-s mark exhaled his breath, and then breathed normally thereafter. Figure 6 shows this unprocessed waveform.* Note that the first and second heart sounds are clearly discernable for the first 15 s, but are almost completely masked during breath events (after the 15-s point). Figures 7 and 8 show the chirp-z spectrogram computed from 0 to 750 Hz of this waveform, starting 5 s before and ending 10 s after the first breath was released. In order to better resolve the *frequency* spectrum of the acoustic sensor heart sound data in the absence of any interfering breath sounds, a chirp-z spectrogram was calculated from 0 to 400 Hz using a window length that spanned the first approximately 11 s of the waveform shown in Figure 6. This spectrogram is shown in Figure 9. In order to obtain better *time* resolution of the acoustic sensor heart sound data (again in the absence of any breath sounds), a chirp-z spectrogram was (again) calculated from 0 to 400 Hz over the first approximately 11 s of the waveform shown in Figure 6, using a window length of 128 points. This spectrogram is shown in Figure 10.

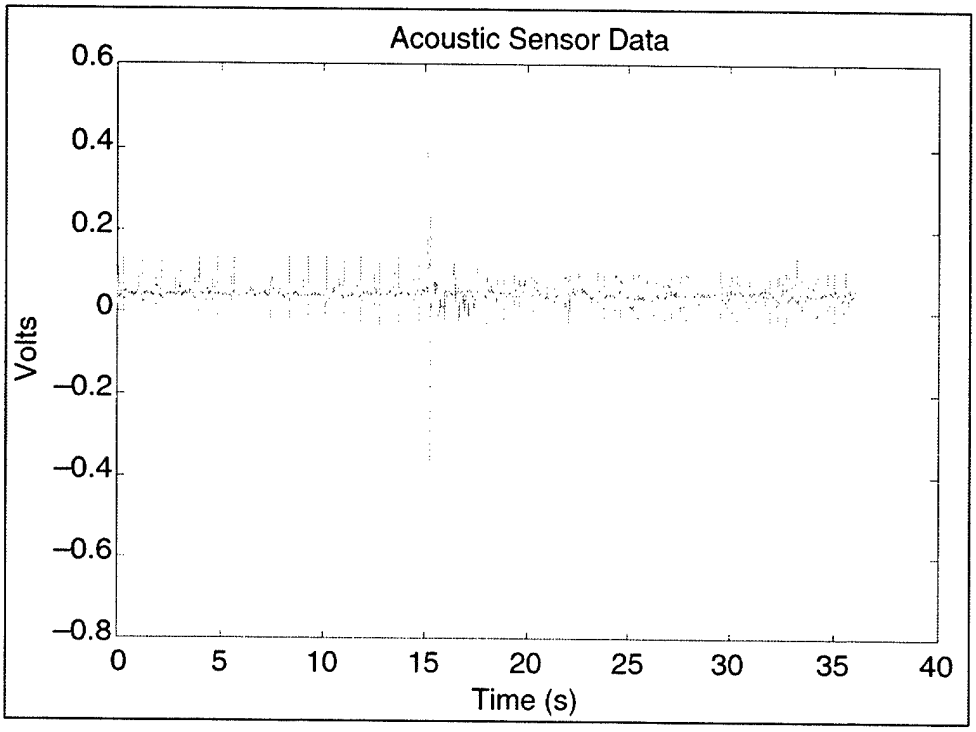

Figure 6. Neck-placed acoustic sensor data taken with subject first holding his breath, then breathing.

Figures 7 and 8 clearly show that the breath sounds are fairly broad band and include low-frequency components that overlap into the spectral region where the heart sounds reside. Note that the uppermost frequencies of the breath sounds were limited by the approximately 520-Hz low-pass analog prefilter circuitry. From Figure 9 it can be observed that most of the spectral energy of the heart sounds occurs at frequencies below approximately 100 Hz. From Figures 9 and 10, it can be observed that the most intense portions of the heart sounds are located between approximately 20 and 60 Hz. Thus, an appropriate filtering strategy would be to eliminate as much of the spectral region (energy) associated with the breath sounds as possible without filtering out too much of the spectral energy of the heart sounds. This was in fact the approach taken for this project; the filter results are detailed in section 5.

*All plots, data analysis, and algorithm development were performed with the use of the Student Version of Matlab, version 5.3.

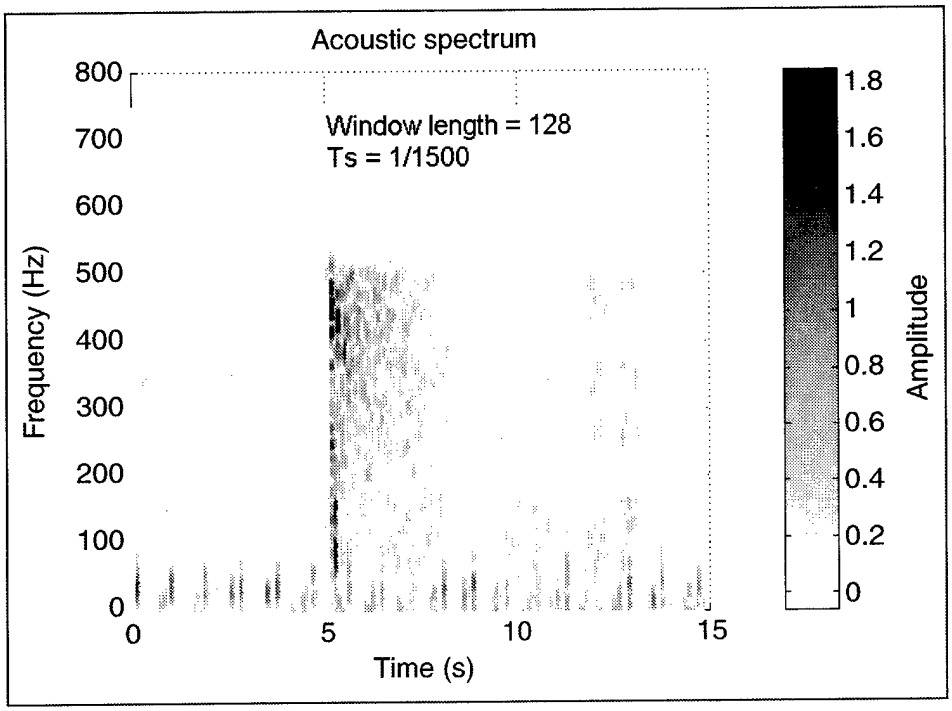

Figure 7. Chirp-z spectrogram of signal shown in Figure 6 (time scale shifted by +10 s).

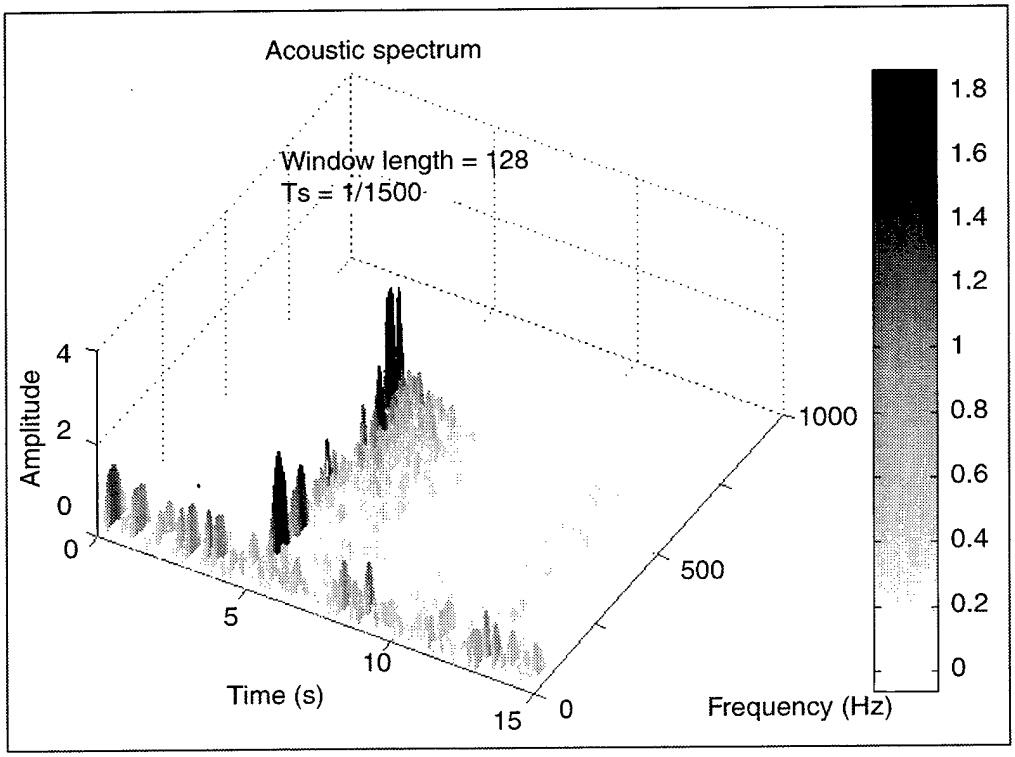

Figure 8. Chirp-z spectrogram (3-D) of signal shown in Figure 6 (time scale shifted by +10 s).

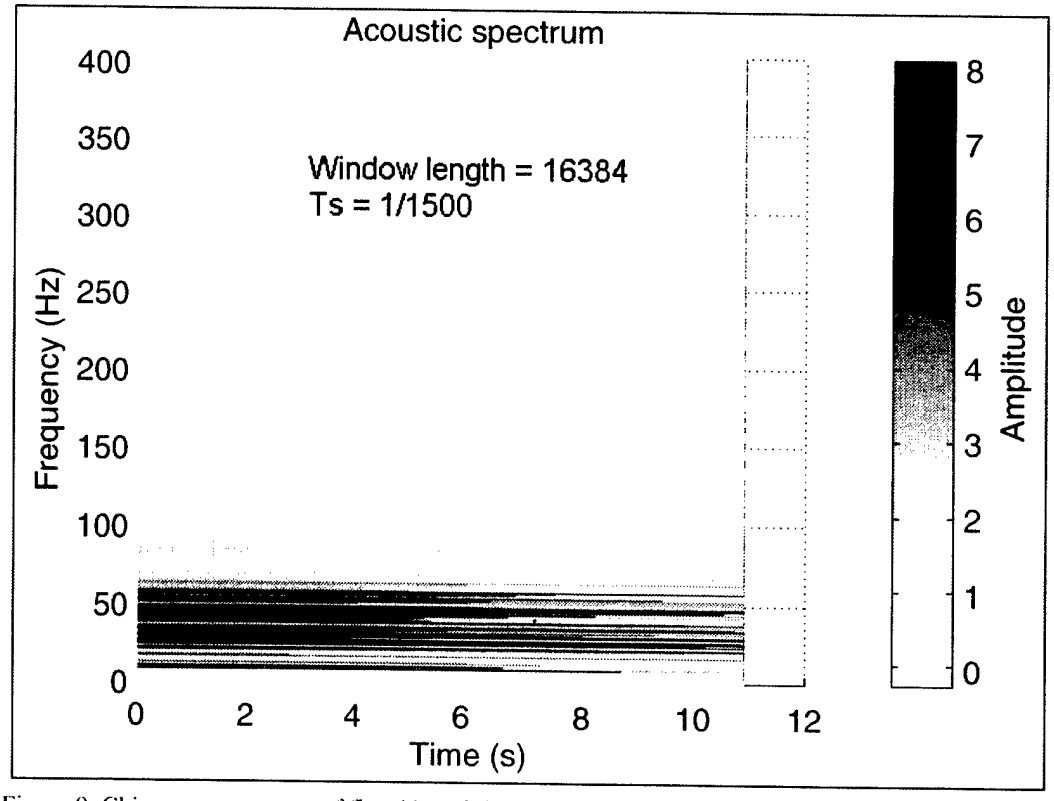

Figure 9. Chirp-z spectrogram of first 11 s of signal shown in Figure 6 using one window of length 16384.

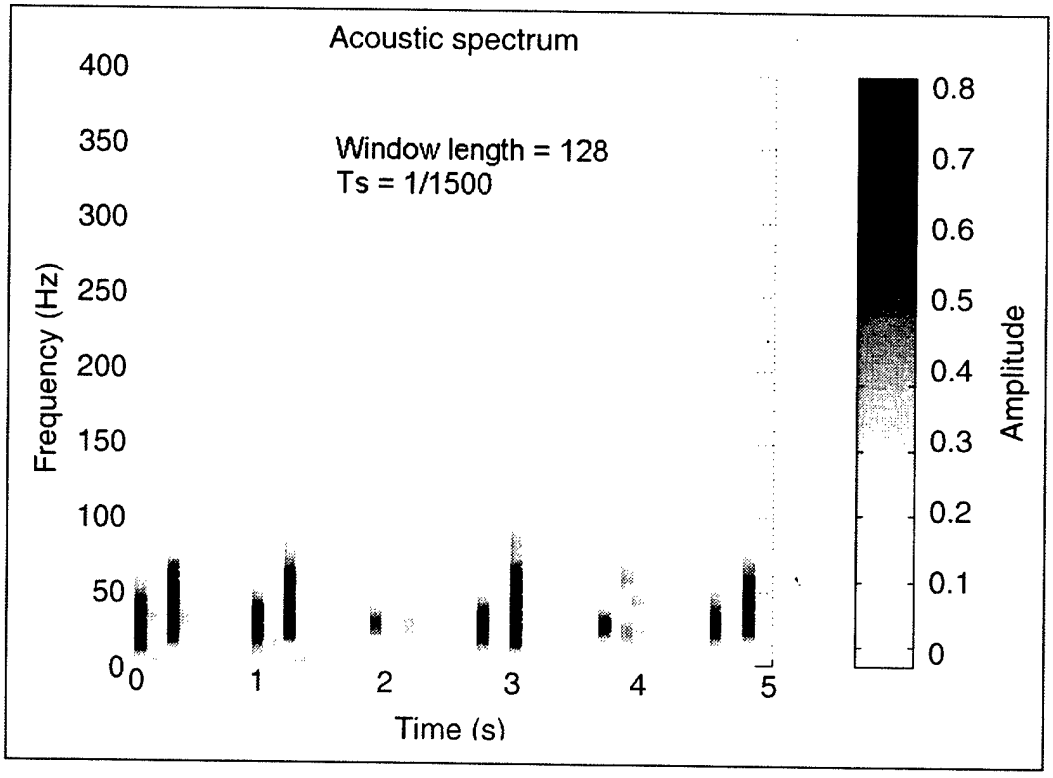

Figure 10. Chirp-z spectrogram of first 5 s of signal shown in Figure 6 using a window length of 128 points.

The spectral analysis of the ECG signals was performed in a fashion similar to that of the acoustic signals. Figure 11 shows a typical captured ECG waveform over an approximately 5-s interval. To obtain a broad view of the *spectral* content of this waveform, a chirp-z spectrogram was calculated from 0 to 750 Hz using a window length of 8192 points. This spectrogram is given in Figure 12. Figure 13 shows a more detailed view and was generated using a frequency range of 0–100 Hz. In order to obtain good *time* resolution of the ECG spectral content of the waveform given in Figure 11, a chirp-z spectrogram was calculated from 0 to 100 Hz using a window length of 128 points. This calculation is displayed in Figures 14 and 15. (Note the presence of 60-Hz noise in all of these ECG spectrograms.)

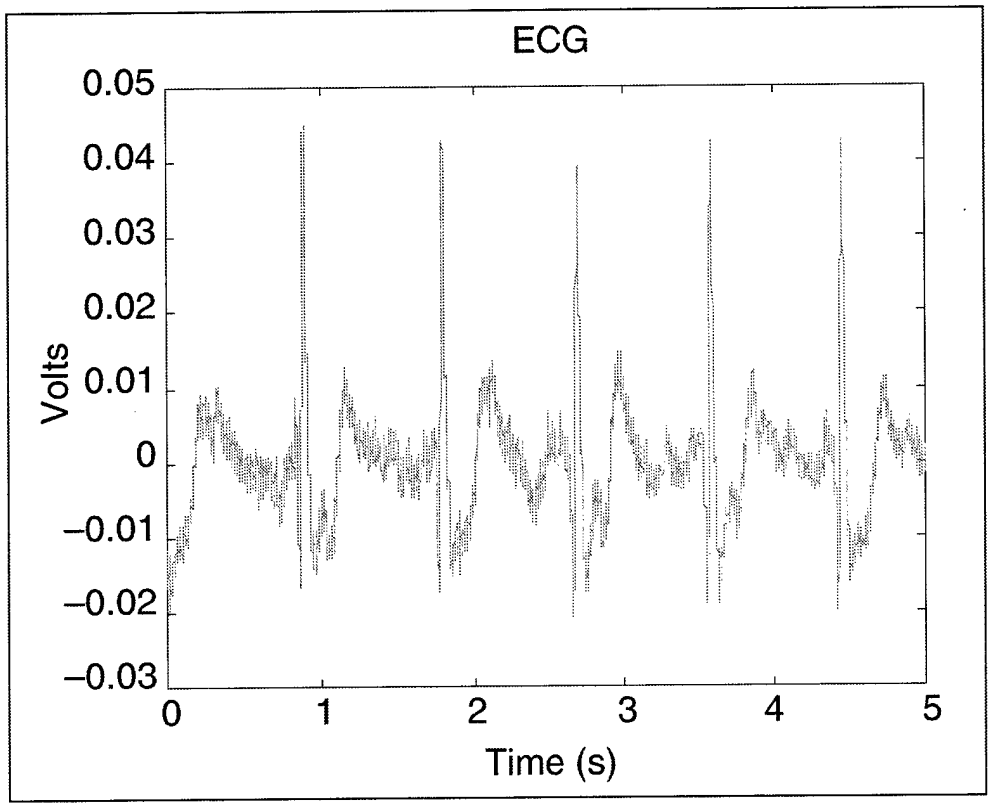

Figure 11. A typical ECG waveform.

Cursory inspection of the ECG spectrogram yields the conclusion that a low-pass filter with a cutoff frequency below 60 Hz would be appropriate and should serve to clean up the signal to a certain degree. The ECG filter results are detailed in section 5.

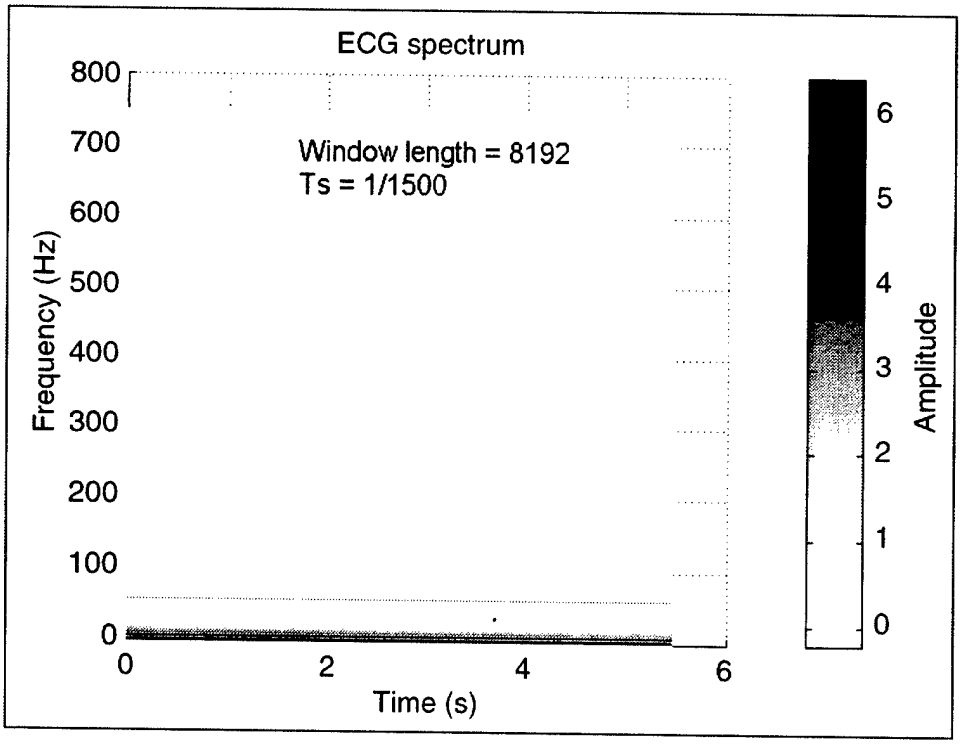

Figure 12. Chirp-z spectrogram of first approximately 5 s of signal shown in Figure 11 using one window of length 8192 from 0 to 750 Hz.

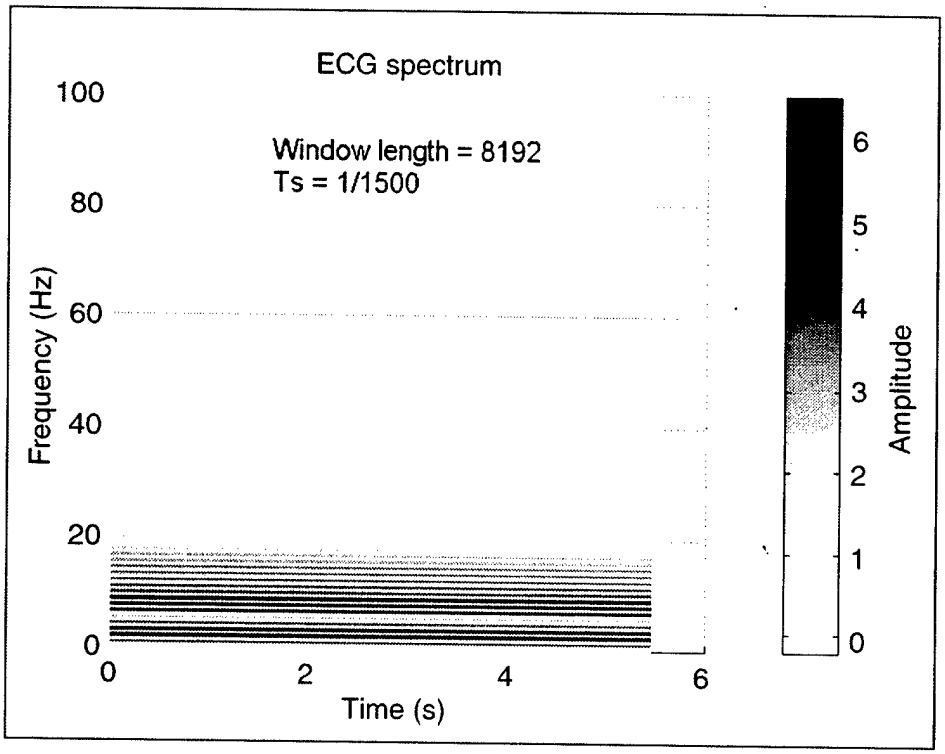

Figure 13. Chirp-z spectrogram of first approximately 5 s of signal shown in Figure 11 using one window of length 8192 from 0 to 100 Hz.

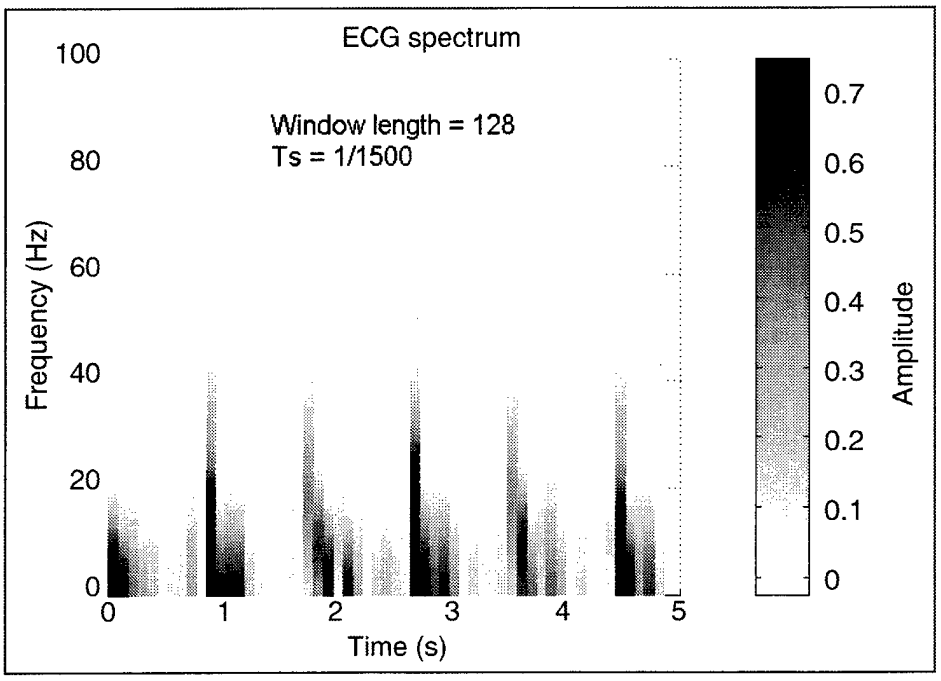

Figure 14. Chirp-z spectrogram of first approximately 5 s of signal shown in Figure 11 using a window length of 128 points.

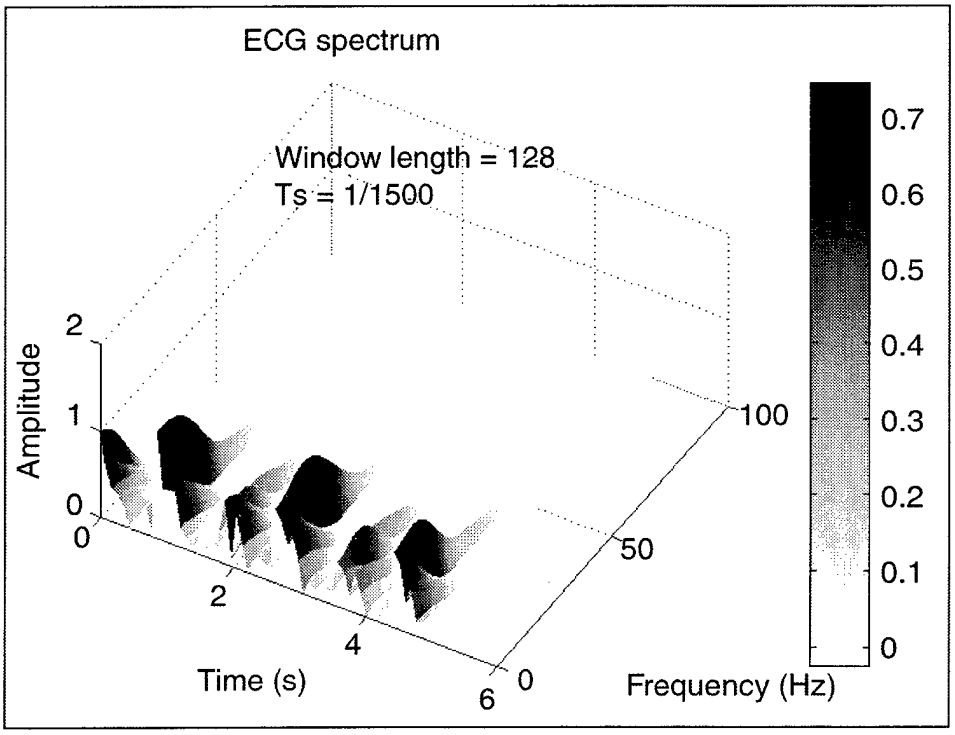

Figure 15. Chirp-z spectrogram (3-D) of first approximately 5 s of signal shown in Figure 11 using a window length of 128 points.

5. Baseline Algorithm

As alluded to earlier, the baseline algorithm fundamentally consists of two processing stages (see Figure 16). The first stage provides the filtering of unwanted spectral components, while the second stage provides the time-domain IBI extraction function. This basic architecture was used for both the acoustic and the ECG IBI extraction algorithms.

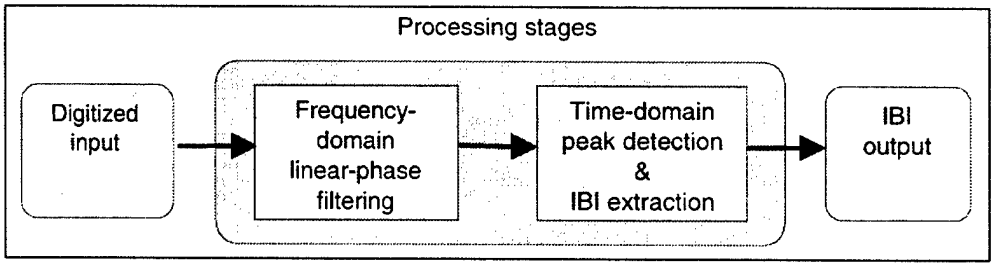

Figure 16. Flow diagram of the heart sound and ECG IBI extraction algorithm.

Armed with the information gleaned from the spectral analysis, several filtering algorithms were evaluated for both the acoustic and the ECG waveforms. In order to ensure a constant group delay for all frequency components of interest and to enable the overall algorithm to time align the filtered waveform with the input waveform, all candidate filters were designed to be of the linear phase type (I). To maximize the efficiency of the filter design, the Park-McClellan optimization algorithm was utilized [8]. All filters were designed to have unity gain in the pass band with –40 dB of rejection in the stop band.

Spectral filters that were designed and evaluated for the acoustic waveforms included a 120-Hz low-pass, a 60-Hz low-pass, a 50-Hz low-pass, a 10- to 50-Hz band-pass, and a 20- to 50-Hz band-pass filter. The filter that appeared to perform the best and the one that was selected for the acoustic filtering section was the 20- to 50-Hz band-pass filter. Figure 17 shows the frequency response and the group delay for the 20- to 50-Hz band-pass filter. Figure 18 shows a portion of the acoustic waveform shown in Figure 6 before and after filtering with the 20- to 50-Hz band-pass filter. Note that the selected filter almost completely removed the breath sound component(s).

For the ECG filtering, a 50-Hz low-pass, a 40-Hz low-pass, and a 10- to 50-Hz band-pass filter were evaluated. The filter that appeared to perform the best (for subsequent IBI extraction) and the one that was selected for the ECG filtering section was the 10- to 50-Hz band-pass filter. Figure 19 shows the frequency response and the group delay for the 10- to 50-Hz band-pass filter. Figure 20 shows the original unprocessed ECG waveform of Figure 11 before and after filtering with the 10- to 50-Hz band-pass filter.

Although the filter selected for the *acoustic* filtering section did a good job in removing most of the breath sound components, closer inspection of the filtered waveforms revealed that too many in-band interference spikes were present for effective direct time-domain extraction of IBIs. To deal with the presence of multiple, but somewhat randomly occurring, short-duration interference spikes, a time-domain filtering process was conceived and evaluated. This process attempts to take advantage of the burst nature of the heart sounds by applying a sliding root-mean-squared (RMS) power-averaging window.

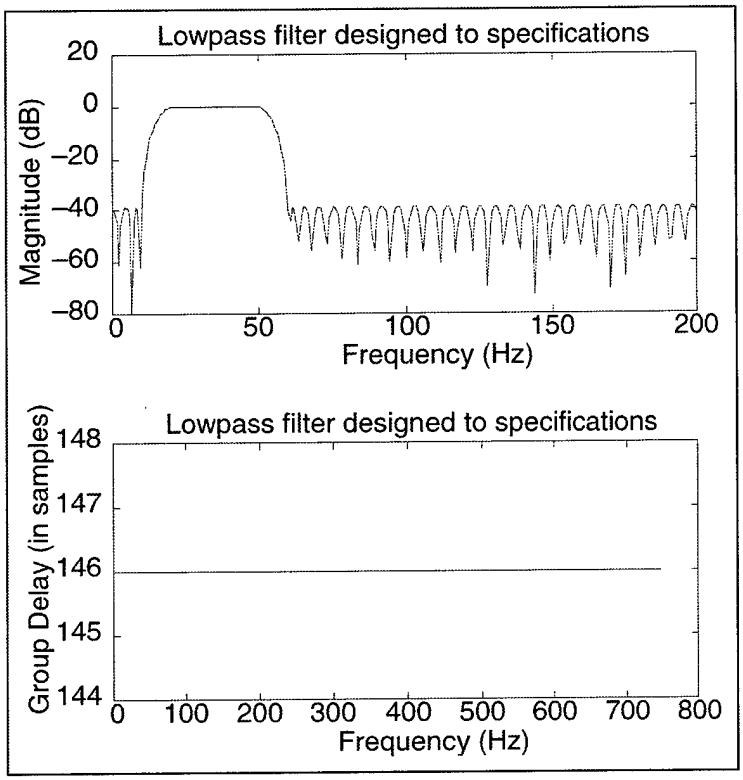

Figure 17. Frequency response and group delay for the 20- to 50-Hz band-pass filter.

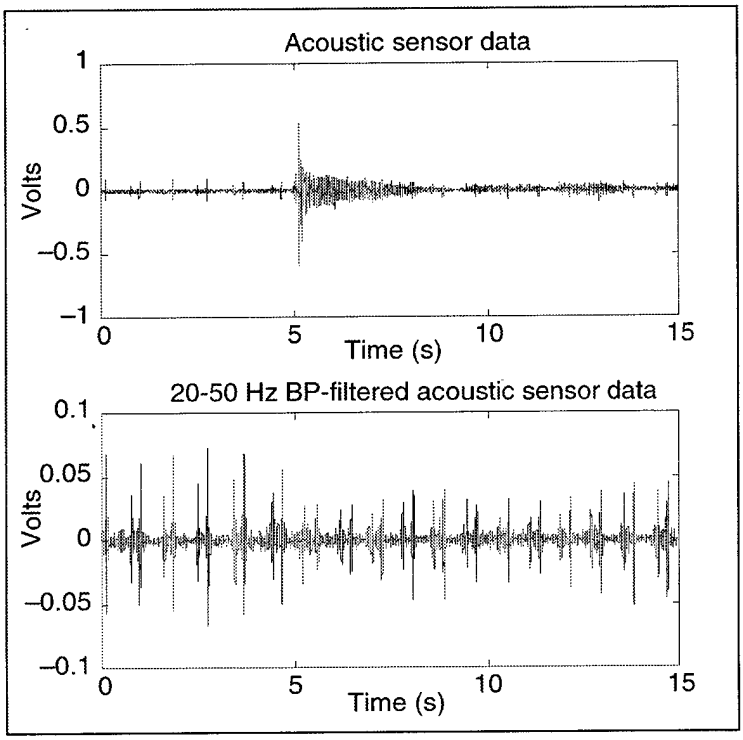

Figure 18. Original (unprocessed) and 20- to 50-Hz band-pass-filtered acoustic waveform.

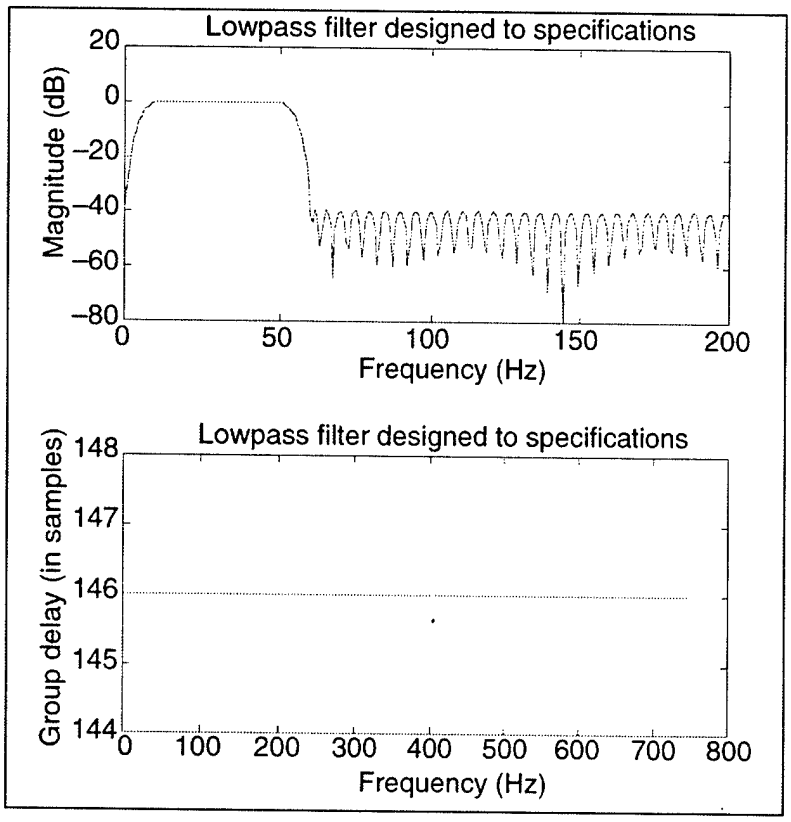

Figure 19. Frequency response and group delay for the 10- to 50-Hz band-pass filter.

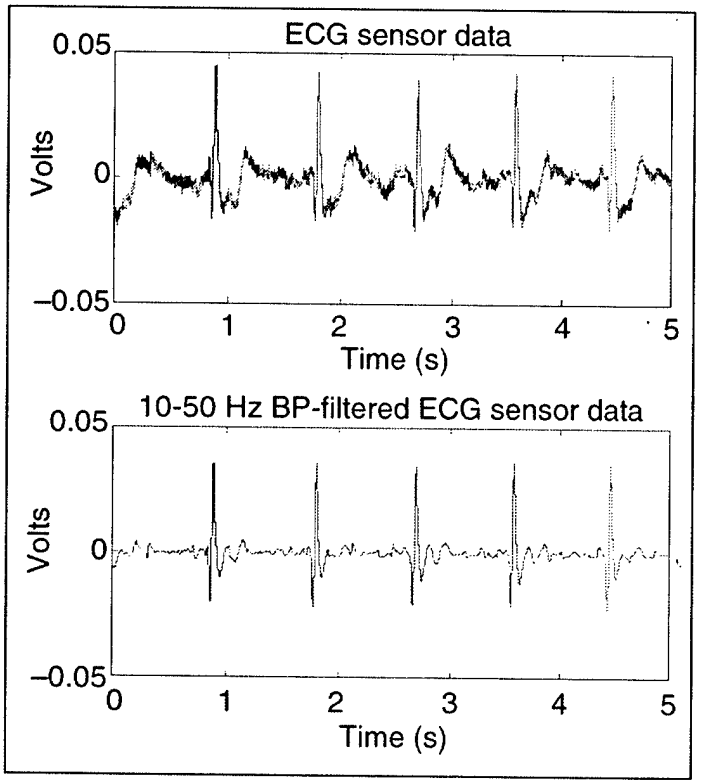

Figure 20. Original (unprocessed) and 10- to 50-Hz band-pass-filtered ECG waveform.

The basic idea of the sliding window was to chose a window length that is approximately equal to the lengths of the first and second heart sounds and to compute an average RMS power within each window. When the window is moved to a location that aligns with the heart sound burst, a relatively high RMS power will be computed (power centroid). When the window is moved away from a heart sound burst, a relatively low RMS power will be computed even if it contains a few short-duration interference spikes. By varying the window length and the amount of window overlap from one RMS power computation to the next, the algorithm can be optimally tuned.

Two RMS power windowing algorithms were evaluated. The first algorithm accumulates a running sum of the average (RMS) power computed for each window position by first filling a temporary window, time-aligned with the sliding window, with the calculated RMS value at that position. The algorithm then adds the temporary window to the running-sum output sequence. The second algorithm simply assigns the computed RMS value to the position associated with the middle of the window and then moves the window to the next position. This algorithm has the inherent property that it creates decimation of the input sequence which proportionally reduces the number of computations required by the rest of the algorithm by the number of sample points skipped with each move of the window (window length minus window overlap). The cost of increasing the number of samples that the sliding window moves between successive RMS computations is a proportional reduction in the basic accuracy of the extracted IBI times.

Figure 21 shows the results of subsequently processing the 20- to 50-Hz band-pass-filtered acoustic waveform given in Figure 18 with each of the two RMS power-shaping algorithms previously described.

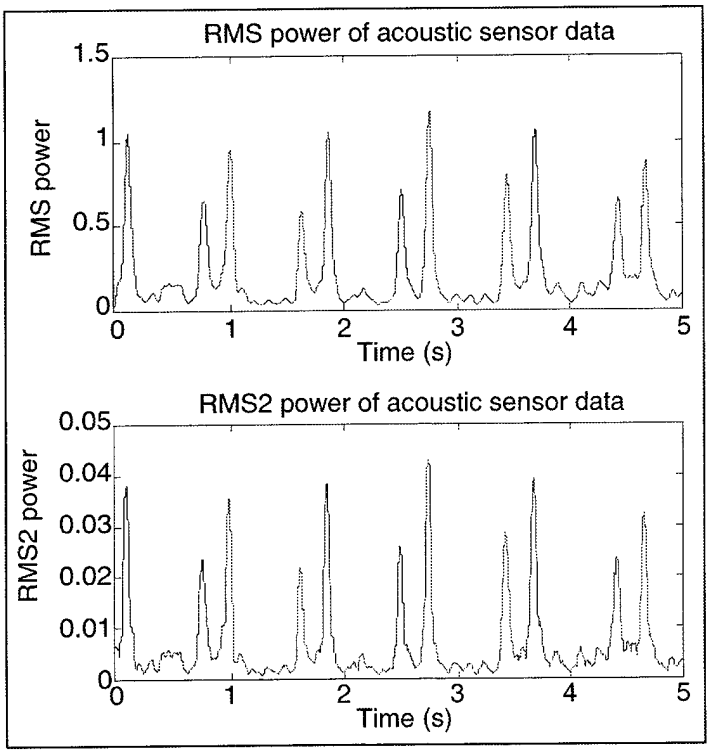

Figure 21. Results of the two RMS power-shaping algorithms on the 20- to 50-Hz band-pass-filtered acoustic waveform given in Figure 18.

The results of combining the initial band-pass filtering with either of the two RMS average power algorithms yields a heart-sound-containing waveform that is *relatively* noise free and reasonably amenable to a peak detection extraction process. For this project, the "running sum" RMS power algorithm was selected as the preconditioning process to the IBI time-domain extraction process because it generally produced less variance around the peaks.

The final portion of the heart-sound time-domain peak detection and IBI extraction stage, like the RMS power-shaping substage, has been designed to use a sliding window approach that sequentially attempts to extract a pair of first and second heart sound IBIs from each window. The extraction algorithm logic has been designed to first determine the appropriate window length, to then establish a peak detection amplitude window, and then to determine the time locations of all peaks within the peak detection amplitude window. Subsequent logic then determines if a *valid* set of IBIs exist within the present window, and then stores the four time locations of the associated first and second heart sounds, if detected. The window is then shifted by a periodically calculated amount, and the process is repeated until the end of the input sequence. A more detailed description of the final portion of the heart sound time-domain peak detection and IBI extraction algorithm is given in Table 1.

Table 1. Final stage of heart sound extraction algorithm.

Step	Description
1	Initialize time window in which to locate the peak response of two heart sound pairs.
2	Determine amplitude window in which to locate the peak response of two heart sound pairs.
3	Locate all heart sound peaks within window.
4	If less than seven peaks are detected, adjust the time window and go back to step 2. Otherwise, evaluate temporal locations of first four heart sounds[a] (no. 1, no. 2, no. 3, and no. 4) to determine if they meet criteria.
5	If first four heart sounds meet criteria, log heart sound times for both pairs of first and second heart sounds. Shift extraction window start point to a time just after second heart sound of second pair. Adjust window length according to criteria. (Go to step 8.)
6	If first four heart sounds do not meet criteria: a) If a set of heart sound pairs was successfully detected and logged from the last pass, check to see if an extra (noise) peak was detected between heart sound peak no. 2 and heart sound peak no. 3, i.e., see if no. 1[a], nos. 2[a] and 4, and no. 5 heart sounds meet criteria: i) If so, log heart sound times for both pairs of first and second heart sounds. Shift extraction window start point to a time just after second heart sound of second pair. (Go to step 8). ii) If not, go to step 7.
7	Check to see if first four heart sounds are out of phase by one sound; shift extraction window start point to a time just after no. 1 (or no. 1[a]) heart sound.
8	Repeat steps 2–7 until the end of the signal stream.

[a]If a set of heart sound pairs was successfully detected and logged from the last pass, heart sounds nos. 3 and 4 of the last (detected) set are substituted for heart sounds nos. 1 and 2 of the present set, respectively.

The specific criteria developed for determining whether a valid set of IBIs is present within a given extraction window is given in Table 2. The criteria, along with a particular logic flow, has been designed to determine if any one of the four hearts sounds was not successfully detected or if any part of a single additional noise perturbation was detected in the region of interest during the peak detection process. The extraction algorithm logic has the ability to identify and reject any single noise perturbation occurring in the region spanned by the four valid heart sounds.

Table 2. Criteria used for determining the presence of a valid set of first and second heart sounds.

Criteria no.	Criteria description
1	P1_P2 < P2_P3
2	P3_P4 < P2_P3
3	P3_P4 < P4_P5
4	P5_P6 < P4_P5
5	P1_P3 < Interval = 1/(20 beats/min)
6	P1_P3 > Interval = 1/(240 beats/min)
7	P2_P4 < Interval = 1/(20 beats/min)
8	P2_P4 > Interval = 1/(240 beats/min)

Note: The terminology P1_P3 means the time interval between the first peak detected in the peak extraction algorithm and the third peak detected, etc.

The second stage of the ECG heart beat-interval extraction algorithm was designed to be essentially identical to the acoustic IBI extraction algorithm, with the notable differences that: the RMS power-shaping process was not used and the only criteria used for determining whether a valid pair of R-to-R ECG peaks were present within a given extraction window, was to check to see if the associated heart beat rate was in the range of 20–240 beats/min.

The full Matlab-based heart sound IBI and ECG heart rate extraction algorithm is listed in the Appendix.

6. Experimental Description

In order to test the heart sound extraction algorithm, an experiment was devised in which two male human subjects (separate trials) were situated alone in a quiet, dark room while watching a recently released, and never-before-seen-by-the-subjects (approximately 2-hr-long) action/adventure video. Prior to the start of the movie, each subject was instrumented with an ARL-designed and constructed acoustic sensor/measurement system and a Holter Win P-V ECG diagnostic monitoring system.* The acoustic sensor was mounted in contact with the right-front

*Diagnostic Monitoring, a division of Cardinc Sciences, Inc., 16931 Millikan Avenue, Irvine, CA 92606.

side of the subject's neck as illustrated in Figure 2a. One channel of the Holter monitoring system was also routed to the ARL acoustic sensor measurement system for simultaneous capture. The two ECG data streams served as ground truth for the acoustic sensor data. In an effort to provide some control of the experimental conditions, each subject was instructed to minimize vocal sounds, to sit quietly, and to breathe normally throughout the experiment.

7. Results and Discussion

The first acoustic sensor data to be processed by the heart sound extraction algorithm was the data shown in Figure 6. Figures 22 and 23 show the first and second heart sound IBI and the ECG IBI extraction results, while Figures 24–29 show the first and second heart sound IBI and the ECG IBI variability results charted in two different fashions.

Two points about the results shown in Figures 22–29 are made. First, the extracted first- and second-heart sound IBIs generally track well with the ECG ground truth results. Second, during two different intervals, the first-heart sound heart rates either first increase and then decrease, or decrease and then increase. A detailed examination of the original unprocessed heart sound data subsequently revealed that this behavior was real; indeed at the approximately 14- to 15-s point, it was, for example, observed that the first heart sound IBI interval was first smaller than and then larger than the adjacent IBI intervals. This IBI variability is also readily observable in the Poincare plots of Figures 24 and 25.

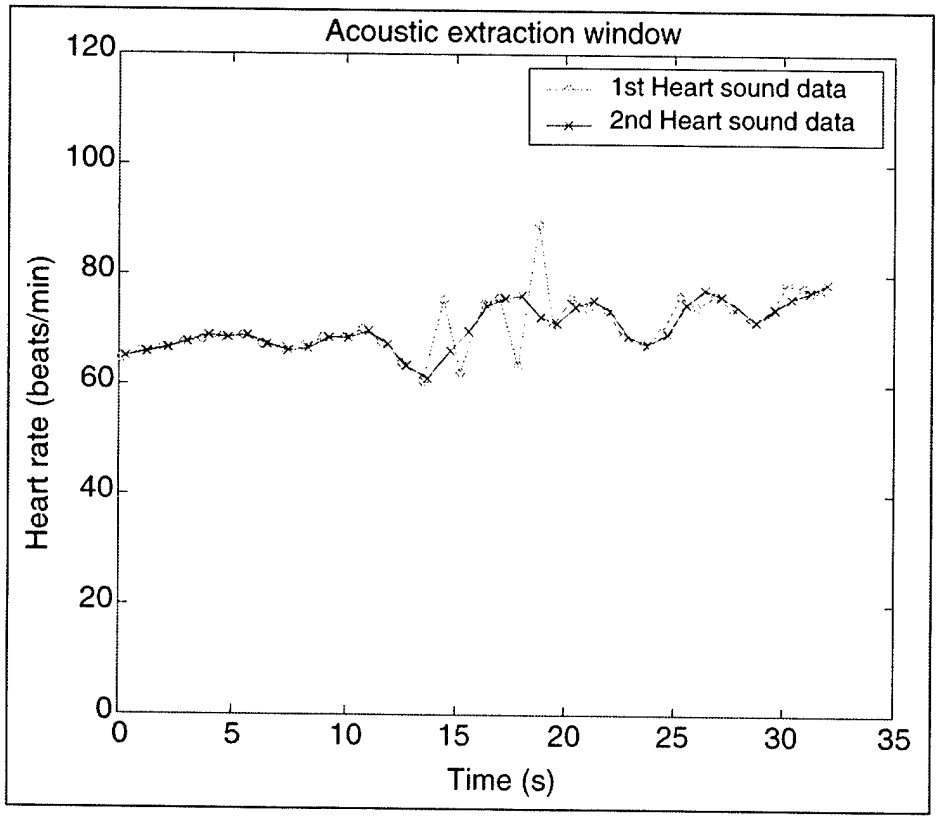

Figure 22. Heart sound extraction results for data shown in Figure 6.

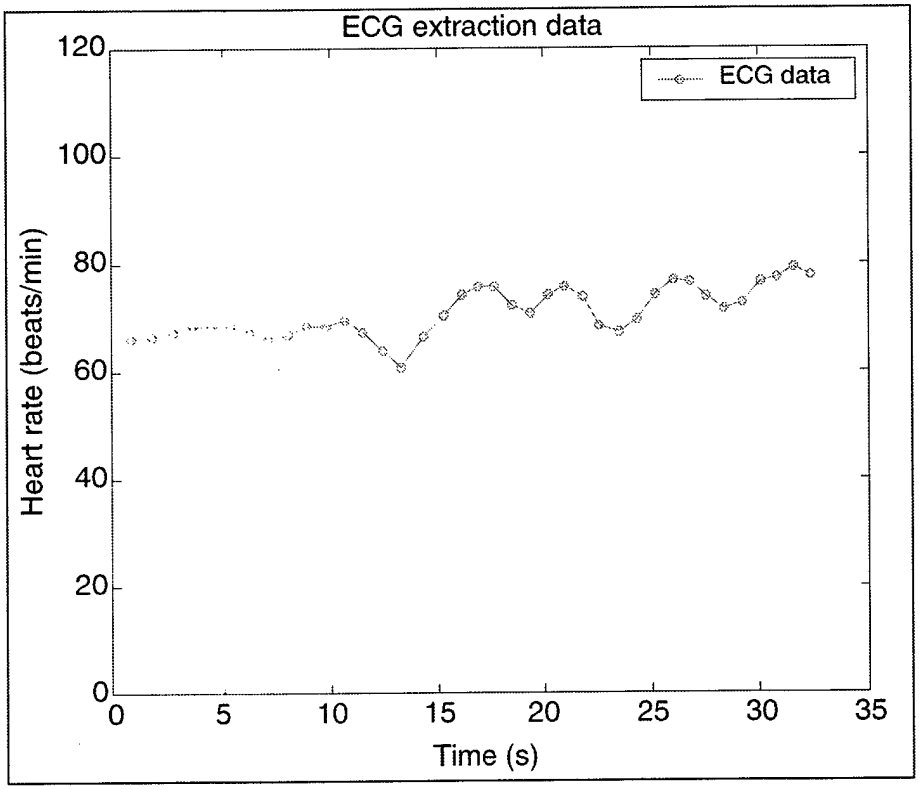

Figure 23. ECG extraction results for data shown in Figure 6.

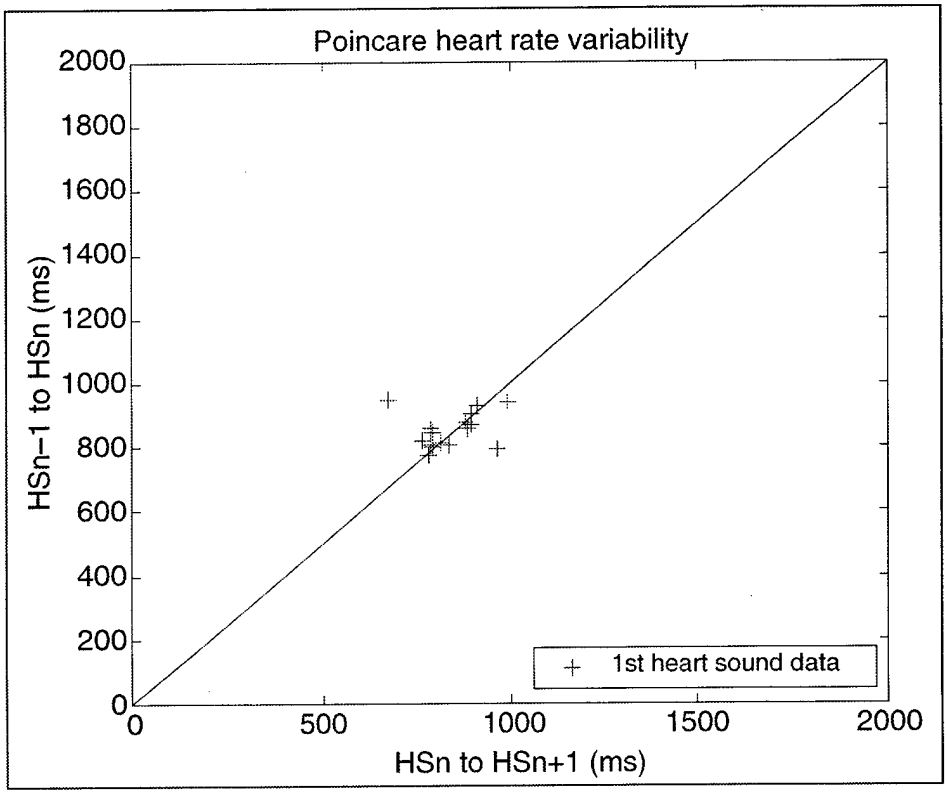

Figure 24. First heart sound variability results for data shown in Figure 6.

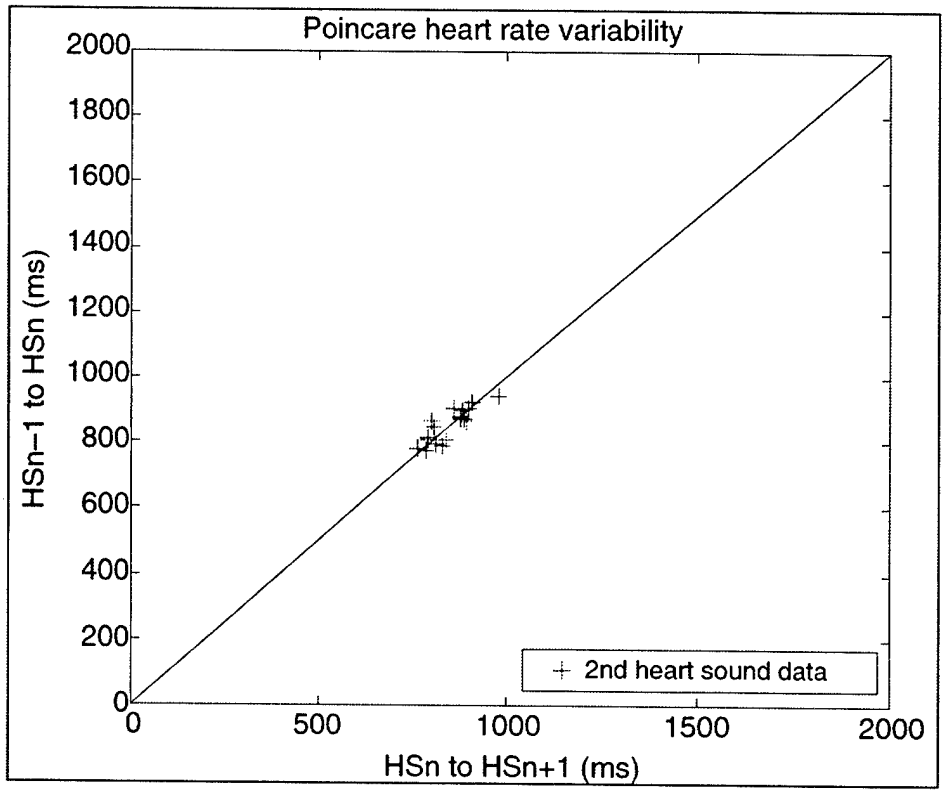

Figure 25. Second heart sound variability results for data shown in Figure 6.

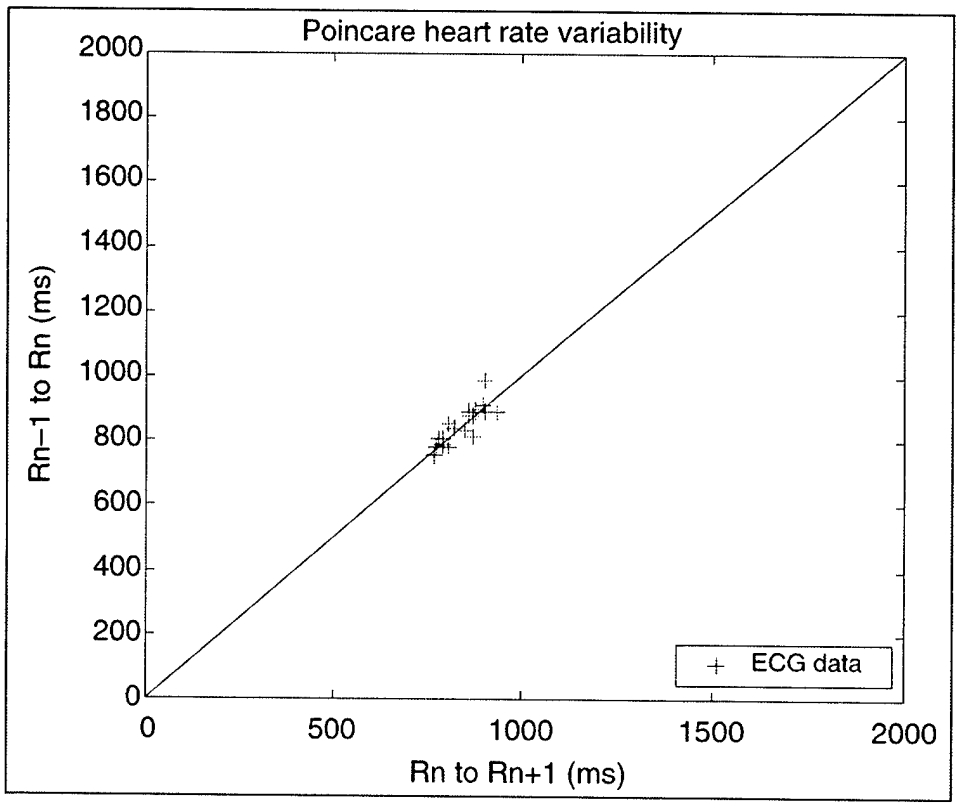

Figure 26. ECG IBI variability results for data shown in Figure 6.

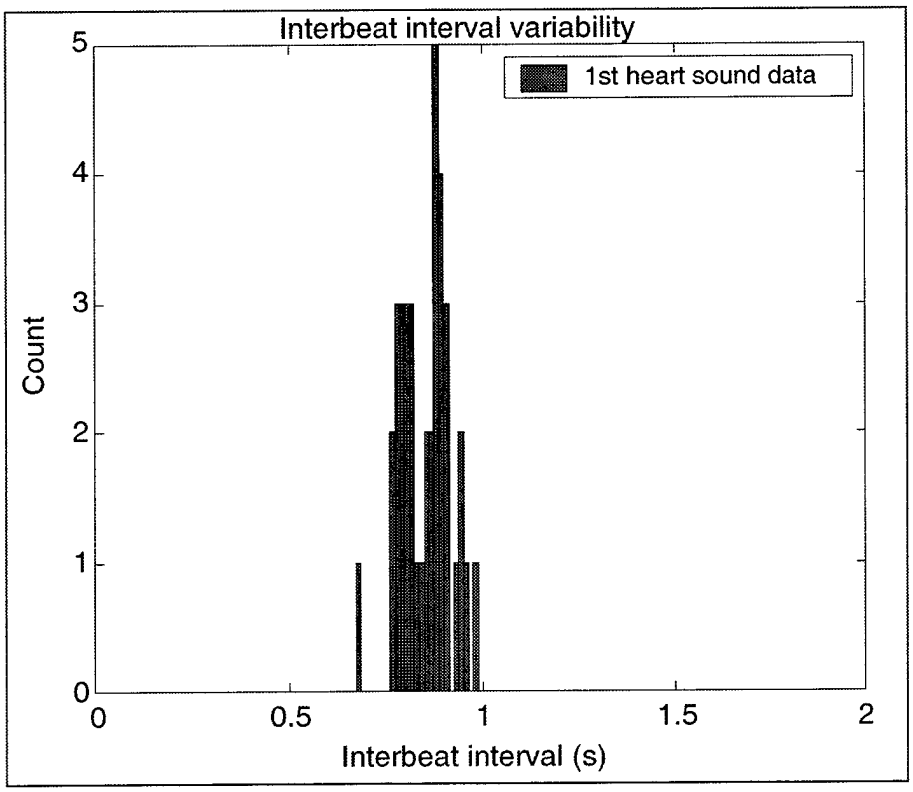

Figure 27. First heart sound variability results for data shown in Figure 6.

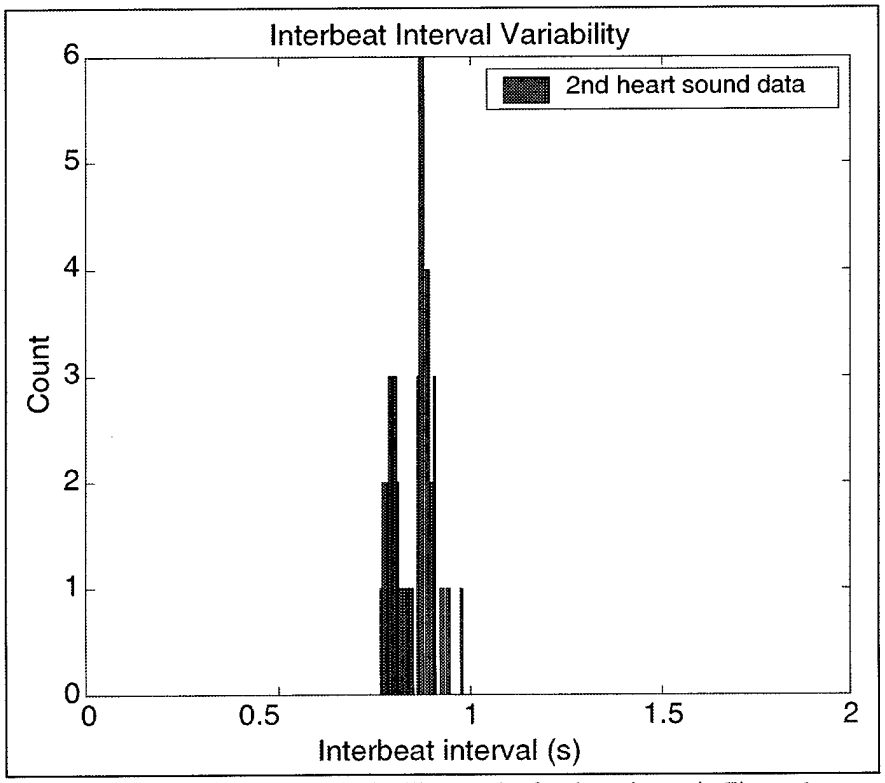

Figure 28. Second heart sound variability results for data shown in Figure 6.

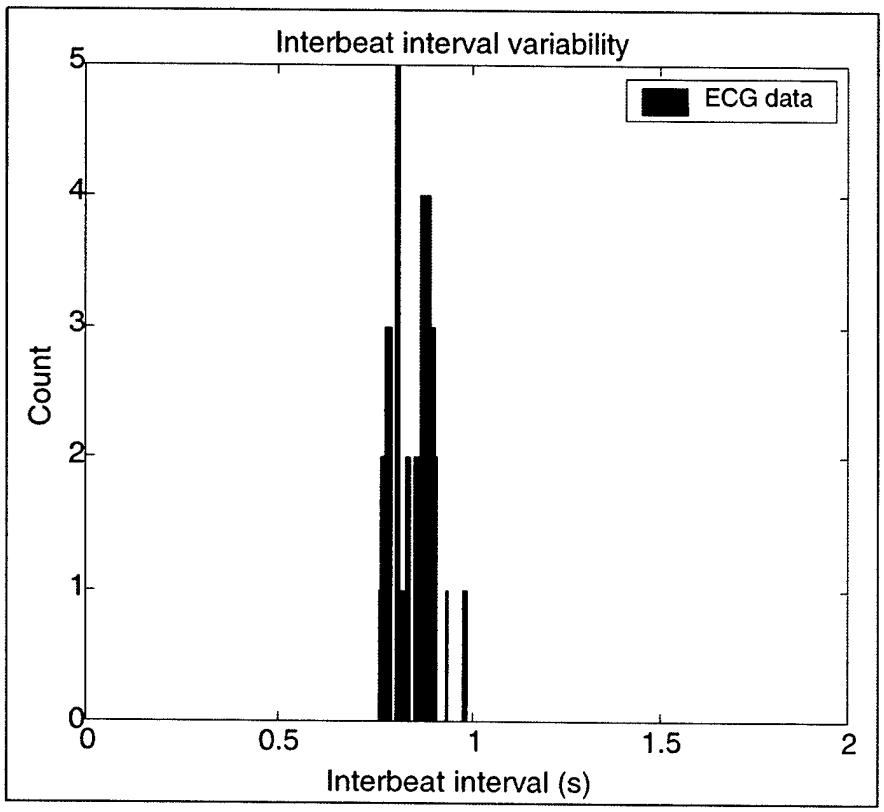

Figure 29. ECG IBI variability results for data shown in Figure 6.

Figures 30–33 show the first and second heart sound IBI and the ECG IBI extraction results for subject no. 1, while Figures 34–41 show the first and second heart sound IBI and the ECG IBI variability results (for subject no. 1), again charted in two different fashions. Note that during this experiment trial, instrumentation trouble developed and only about the first 3300 s of data were recorded.

Comparing the results of Figures 30 and 31 with those of Figures 32 and 33, it is observed that there is again generally good agreement between the first and second heart sound IBI extraction results and the ECG IBI heart rate results. Based on preliminary comparisons of the processed data with the original unprocessed data (subject no. 1), most, if not all of the local perturbations about the trend data appear to be real. It also appears that for those perturbations that significantly vary from the trend, some are real and some represent a failing of the algorithm to either reject extraneous noise in the captured signals or to properly detect all of the valid first and second heart sound pairs. (At the writing of this report, no more-sophisticated diagnostics were available to quantify the reliability and robustness of the heart sound extraction algorithm.) The general variability of the first and second heart sound IBIs (subject no. 1) appear essentially comparable. The distribution of the captured/processed ECG data is noticeably tight.

Figures 42–45 show the first and second heart sound IBI and the ECG IBI extraction results for subject no. 2, while Figures 46–53 show the first and second heart sound IBI and the ECG IBI variability results (for subject no. 2), again charted in two different fashions. Note that during this experimental trial, data was collected for the entire duration of the movie.

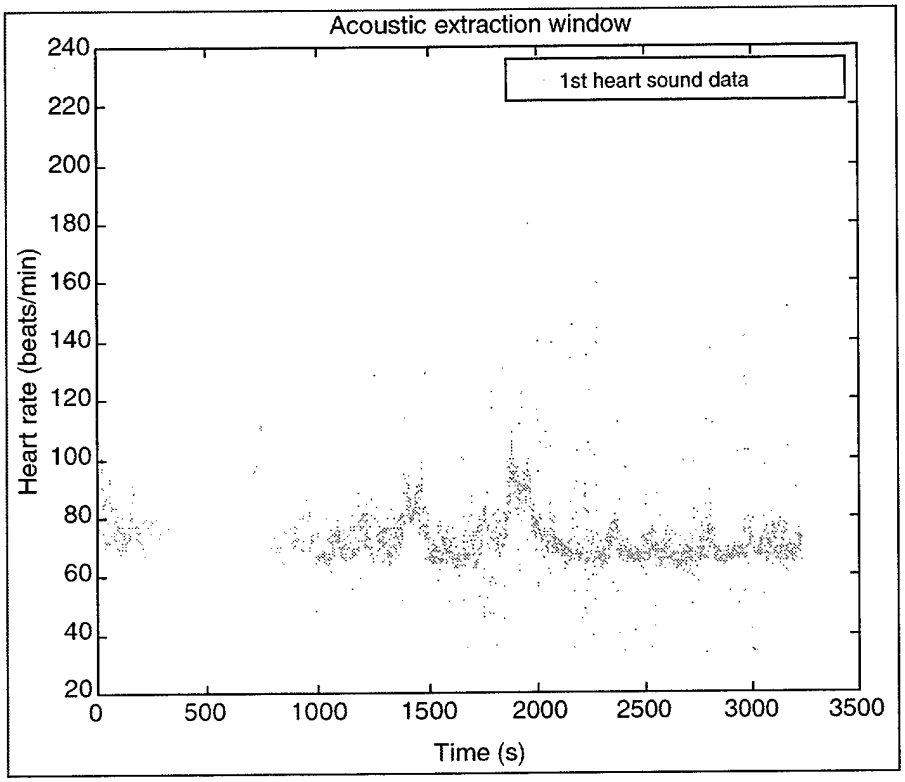

Figure 30. First heart sound extraction results for experimental data taken from subject no. 1.

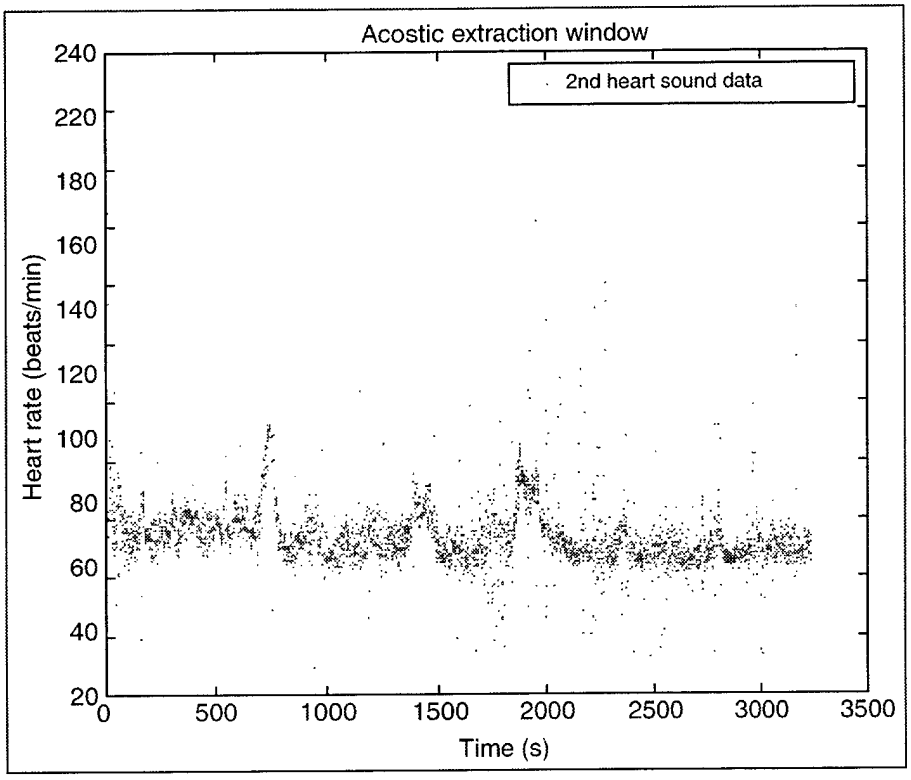

Figure 31. Second heart sound extraction results for experimental data taken from subject no. 1.

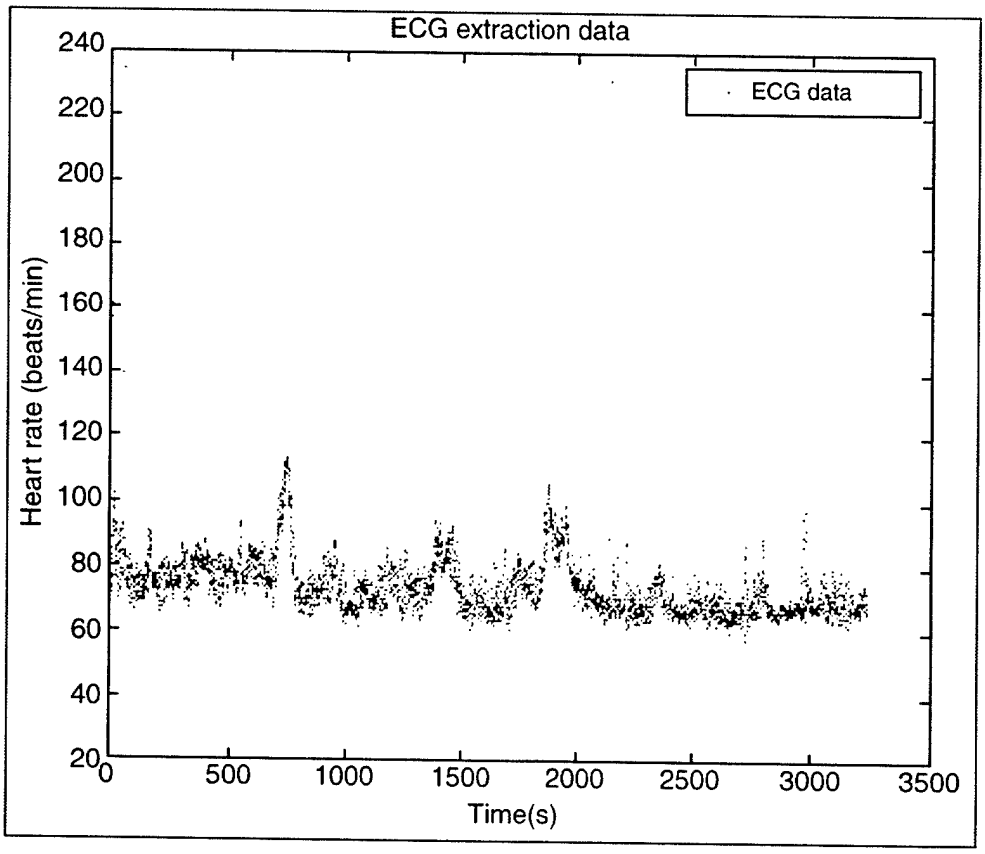

Figure 32. ECG IBI extraction results for experimental data taken from subject no. 1.

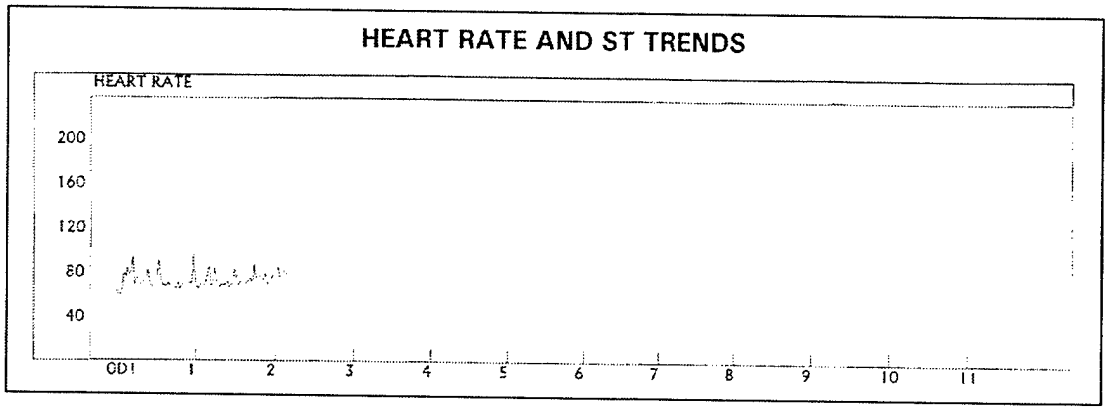

Figure 33. Holter Monitor ECG IBI extraction results for experimental data taken from subject no. 1.

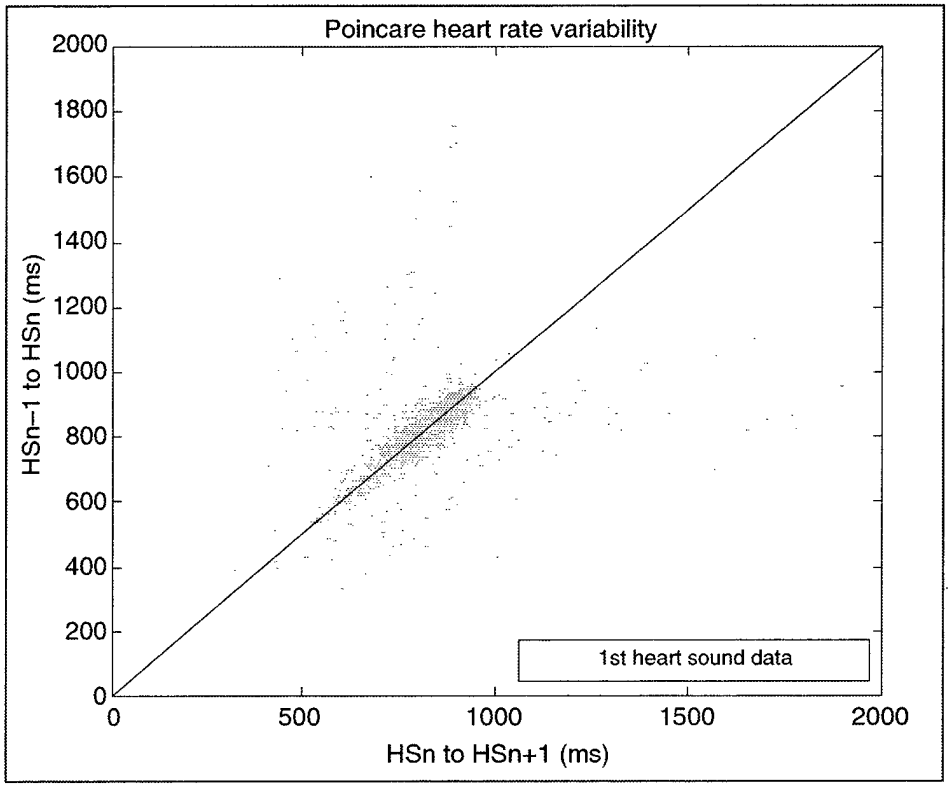

Figure 34. First heart sound variability results for experimental data taken from subject no. 1.

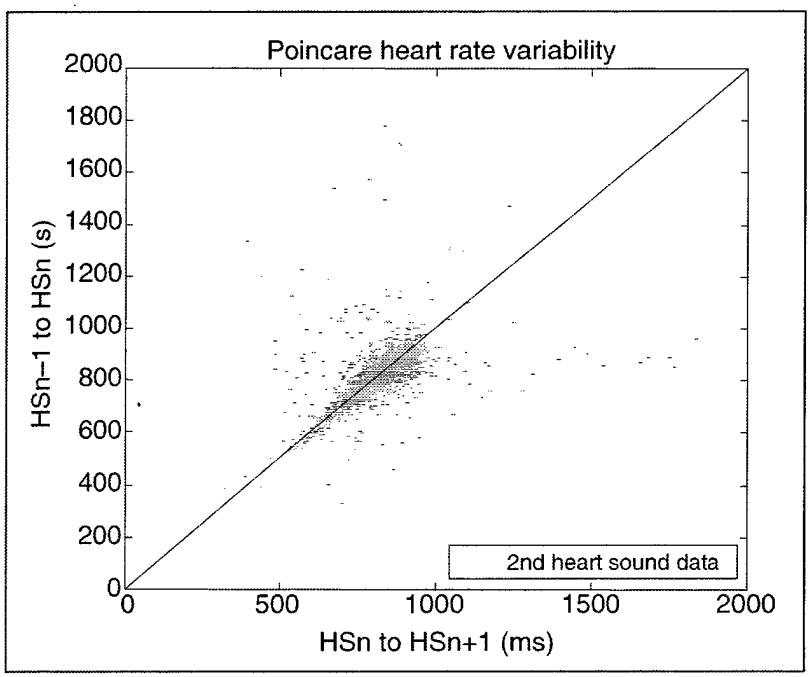

Figure 35. Second heart sound variability results for experimental data taken from subject no. 1.

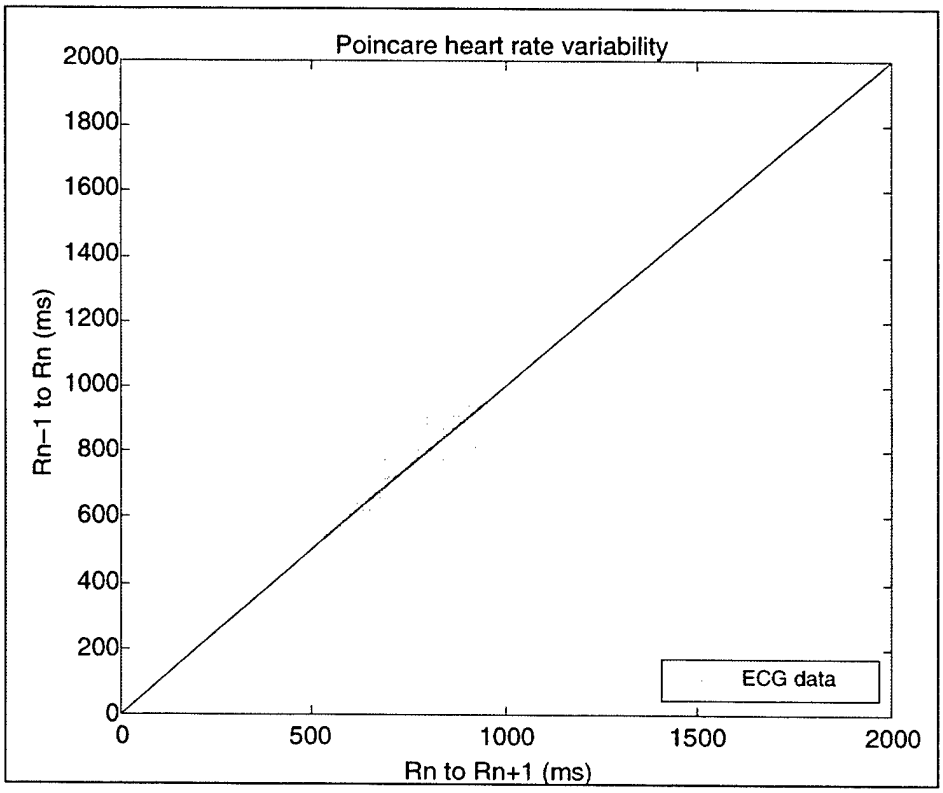

Figure 36. ECG IBI variability results for experimental data taken from subject no. 1.

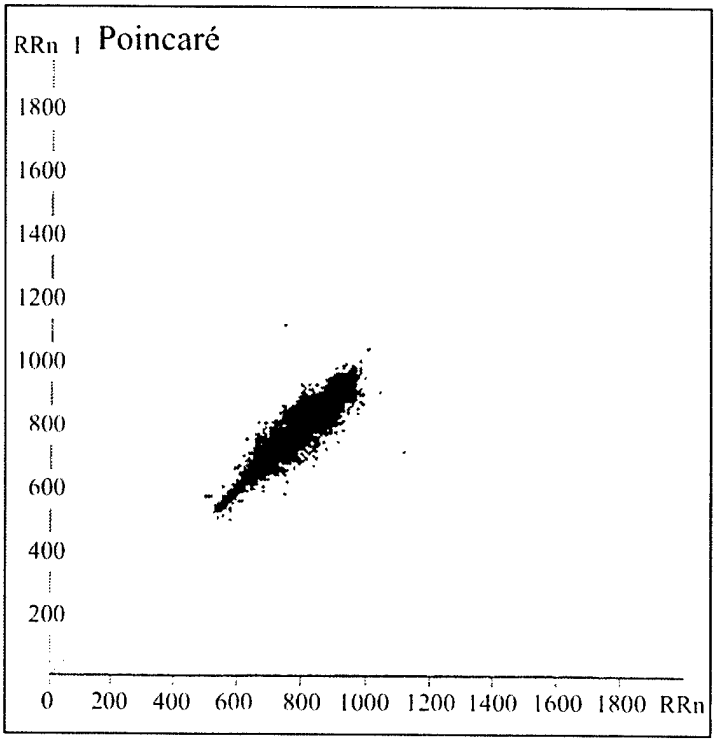

Figure 37. Holter Monitor ECG IBI variability results for experimental data taken from subject no. 1.

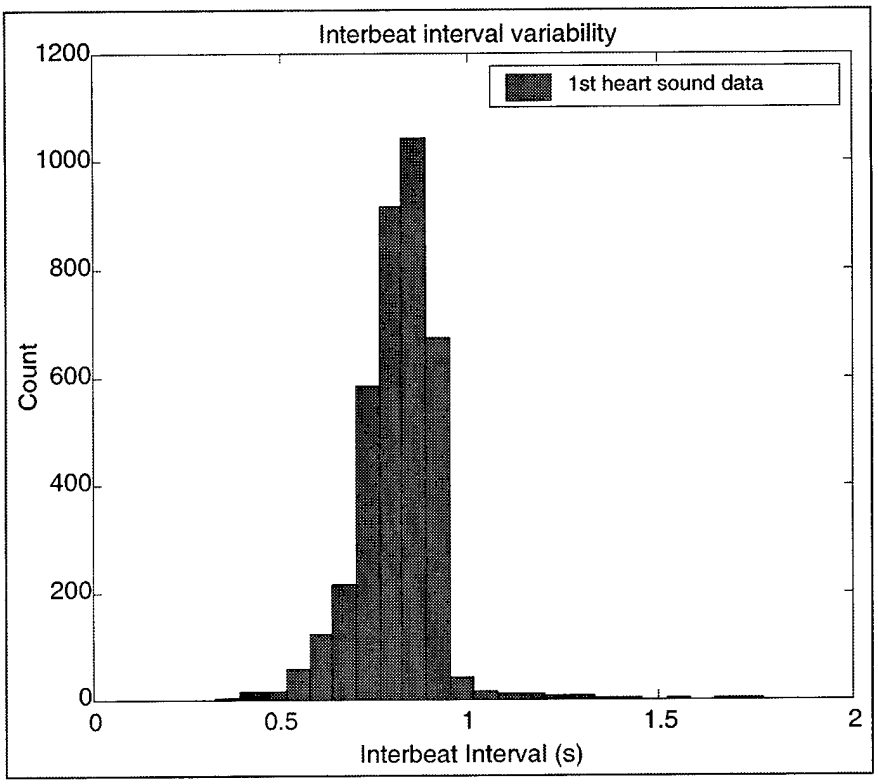

Figure 38. First heart sound variability results for experimental data taken from subject no. 1.

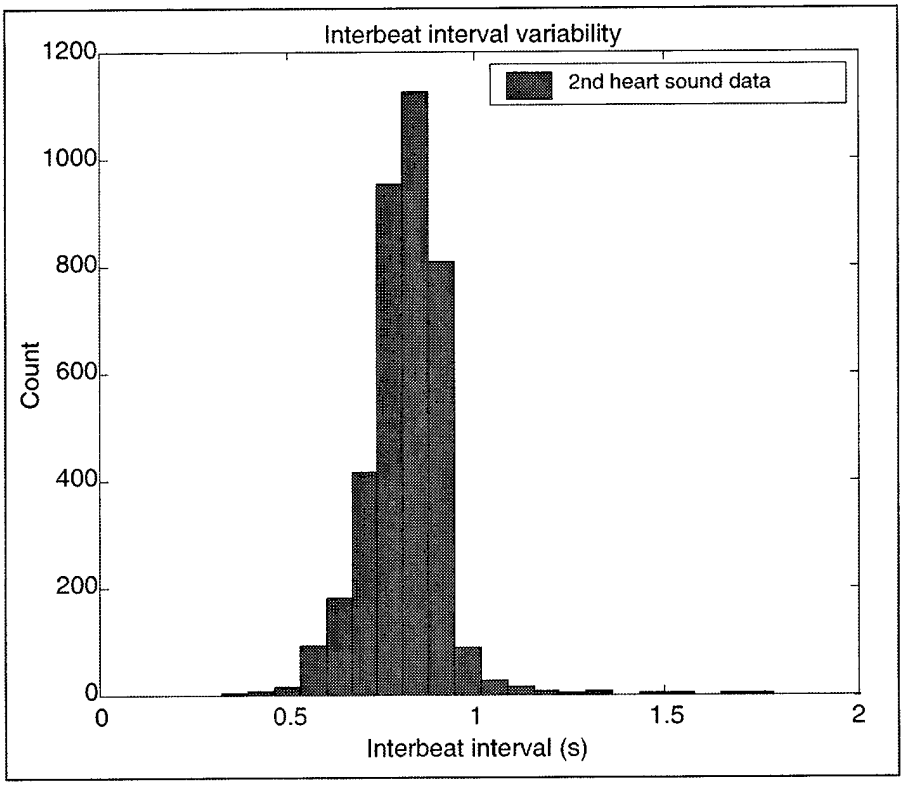

Figure 39. Second heart sound variability results for experimental data taken from subject no. 1.

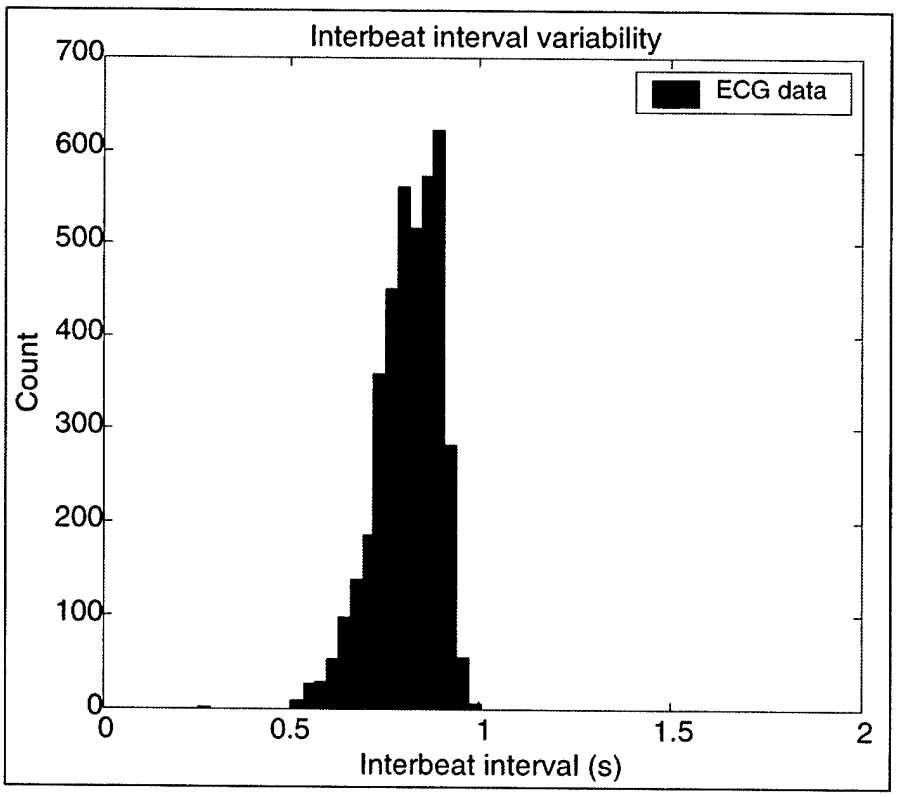

Figure 40. ECG IBI variability results for experimental data taken from subject no. 1.

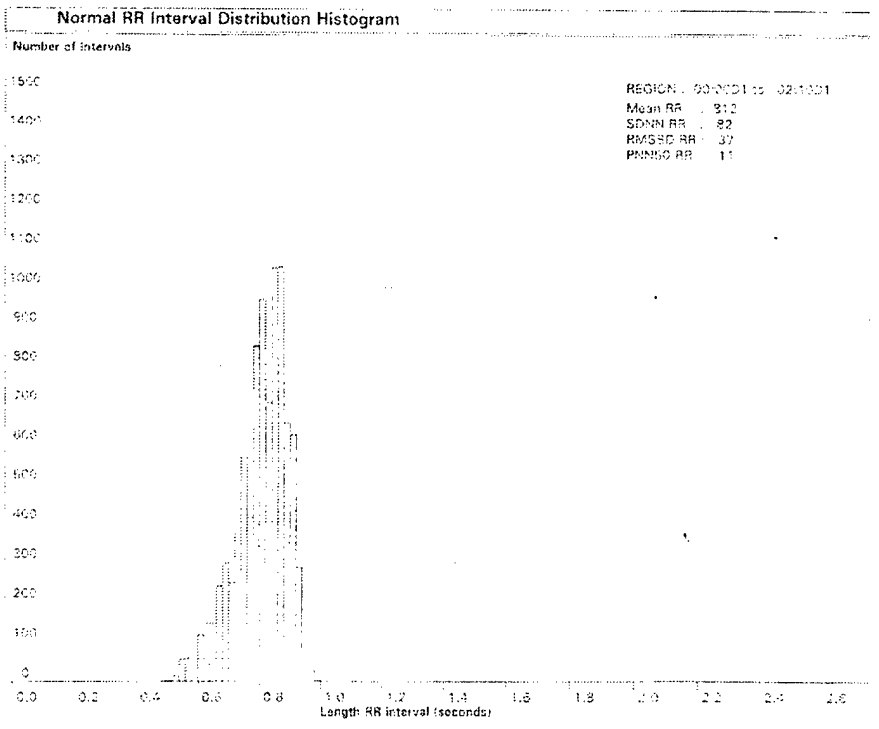

Figure 41. Holter Monitor ECG IBI variability results for experimental data taken from subject no. 1.

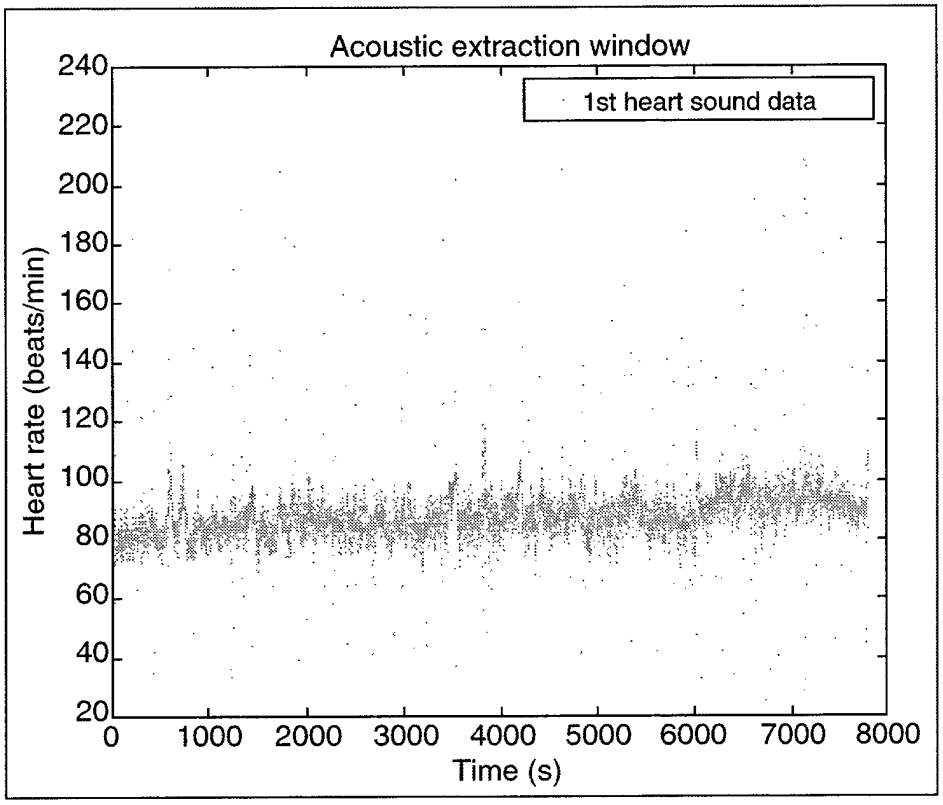

Figure 42. First heart sound extraction results for experimental data taken from subject no. 2.

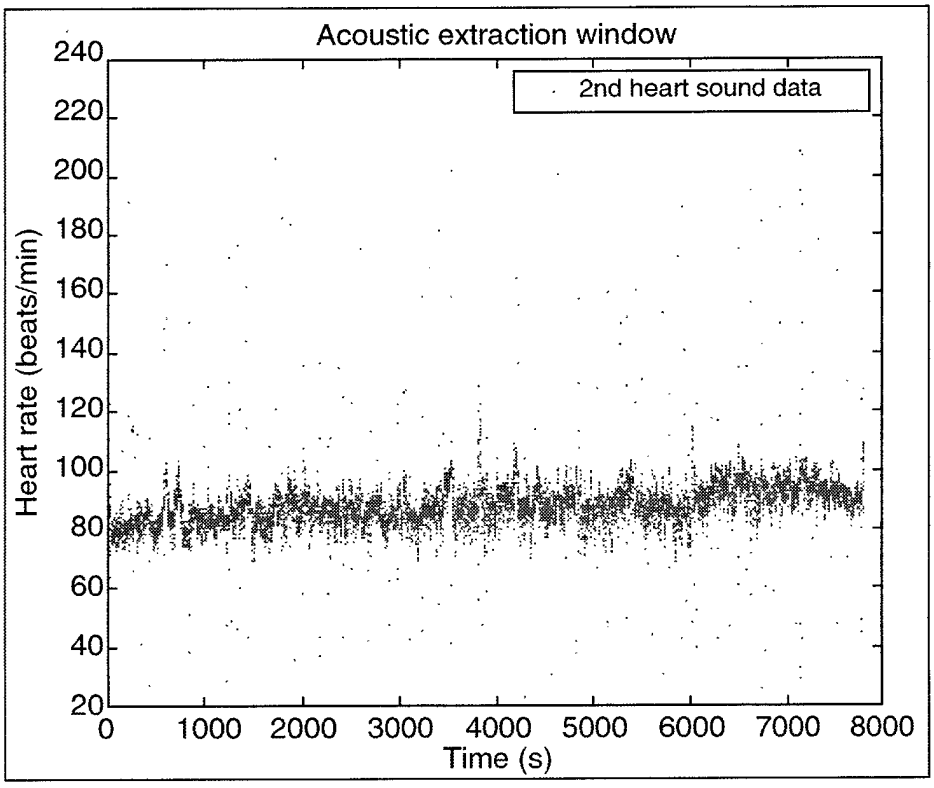

Figure 43. Second heart sound extraction results for experimental data taken from subject no. 2.

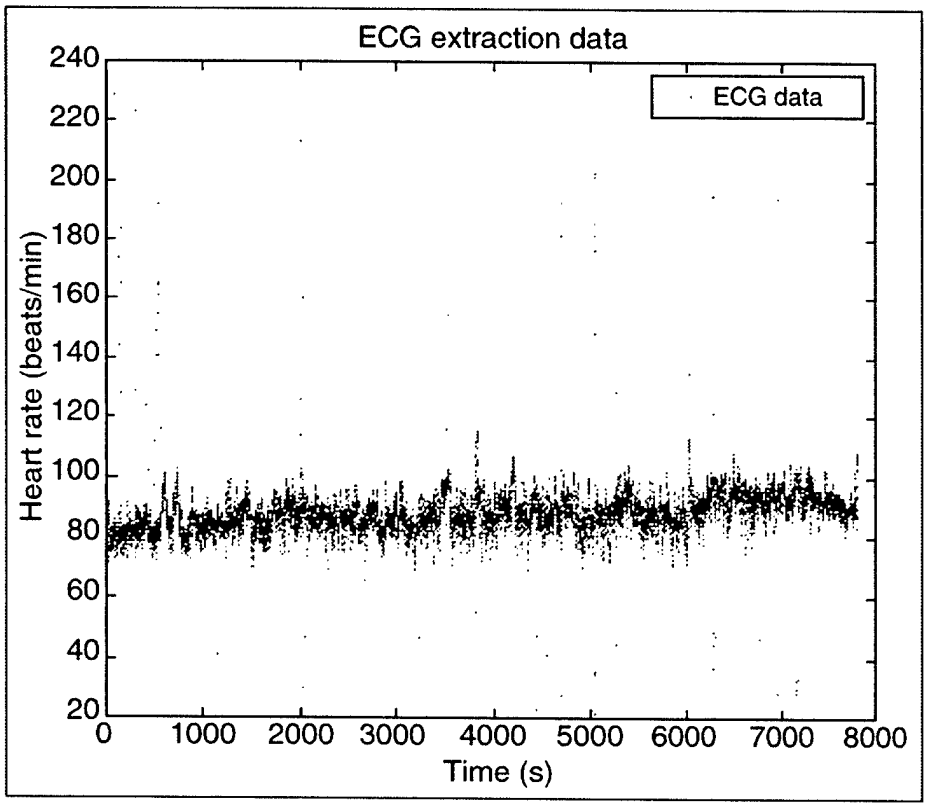

Figure 44. ECG IBI extraction results for experimental data taken from subject no. 2.

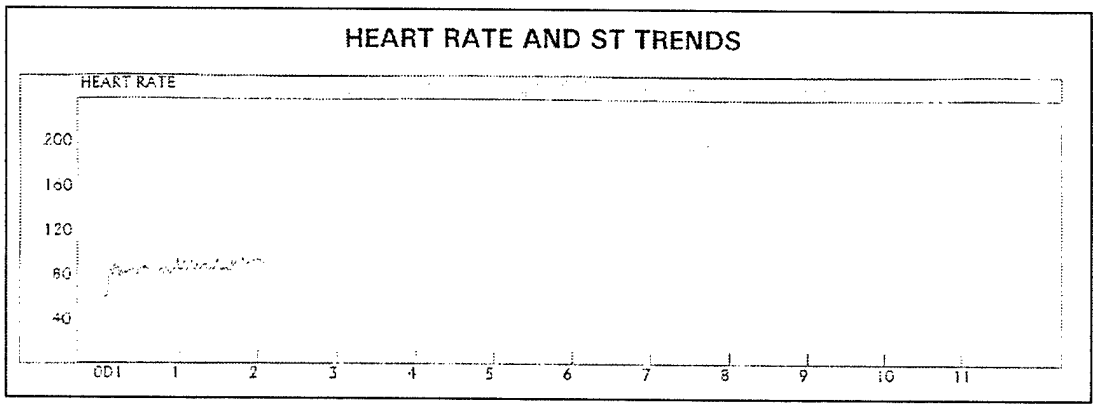

Figure 45. Holter Monitor ECG IBI extraction results for experimental data taken from subject no. 2.

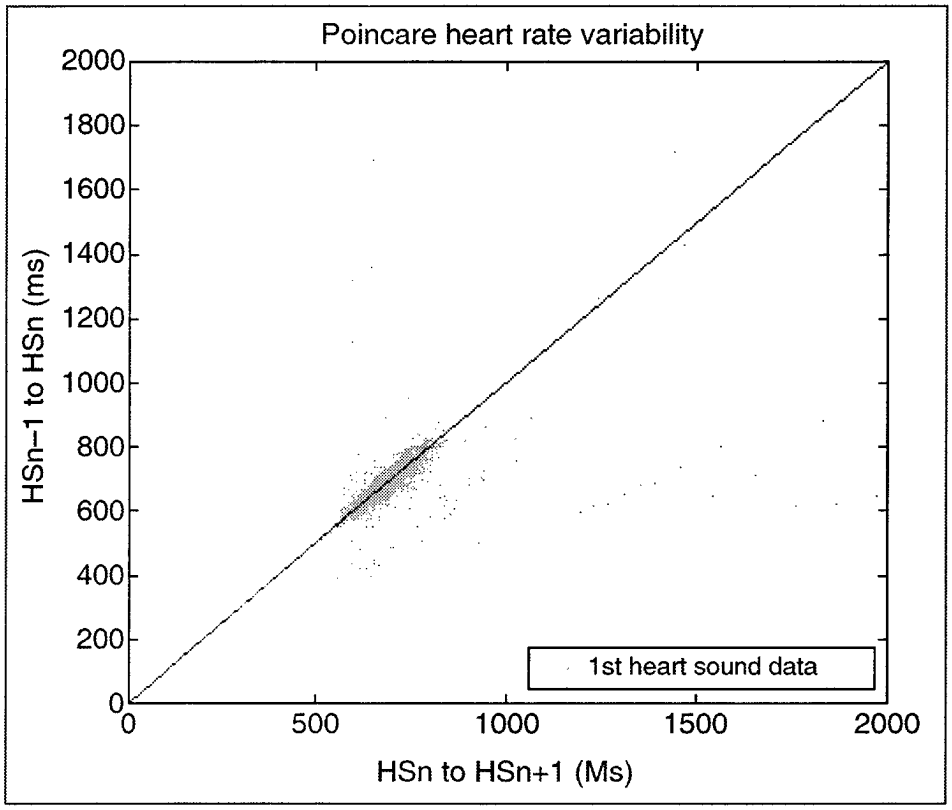

Figure 46. First heart sound variability results for experimental data taken from subject no. 2.

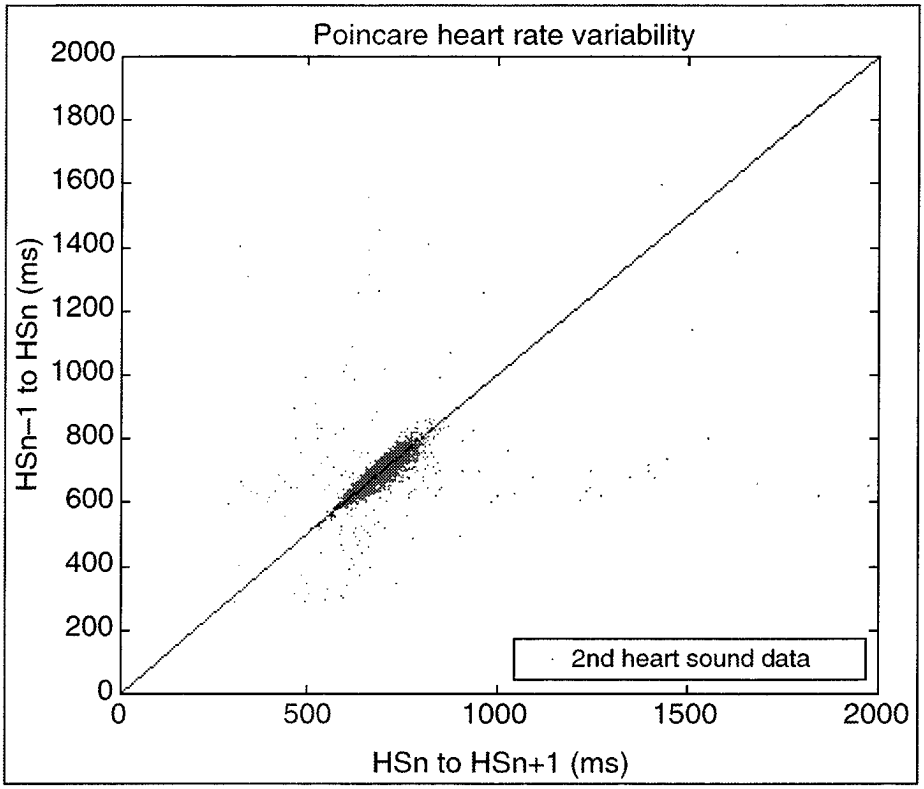

Figure 47. Second heart sound variability results for experimental data taken from subject no. 2.

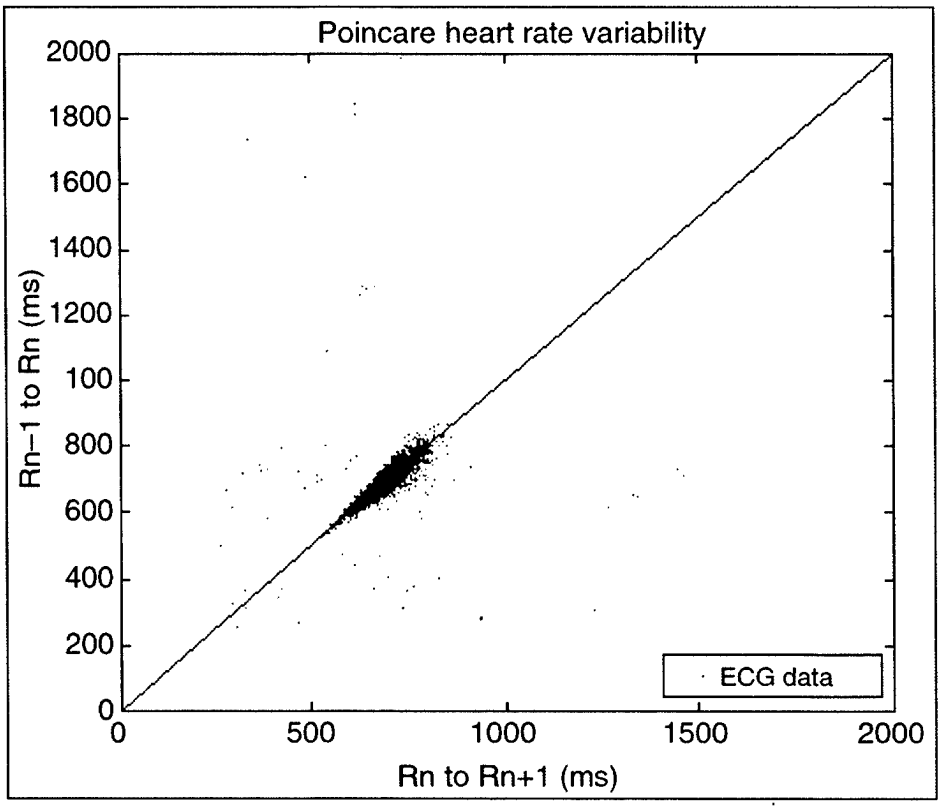

Figure 48. ECG IBI variability results for experimental data taken from subject no. 2.

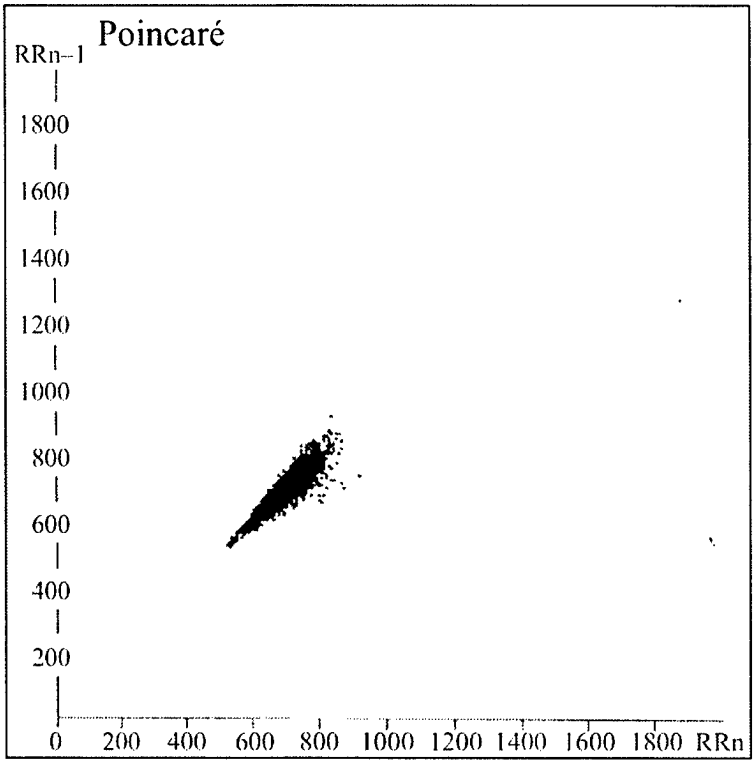

Figure 49. Holter Monitor ECG IBI variability results for experimental data taken from subject no. 2.

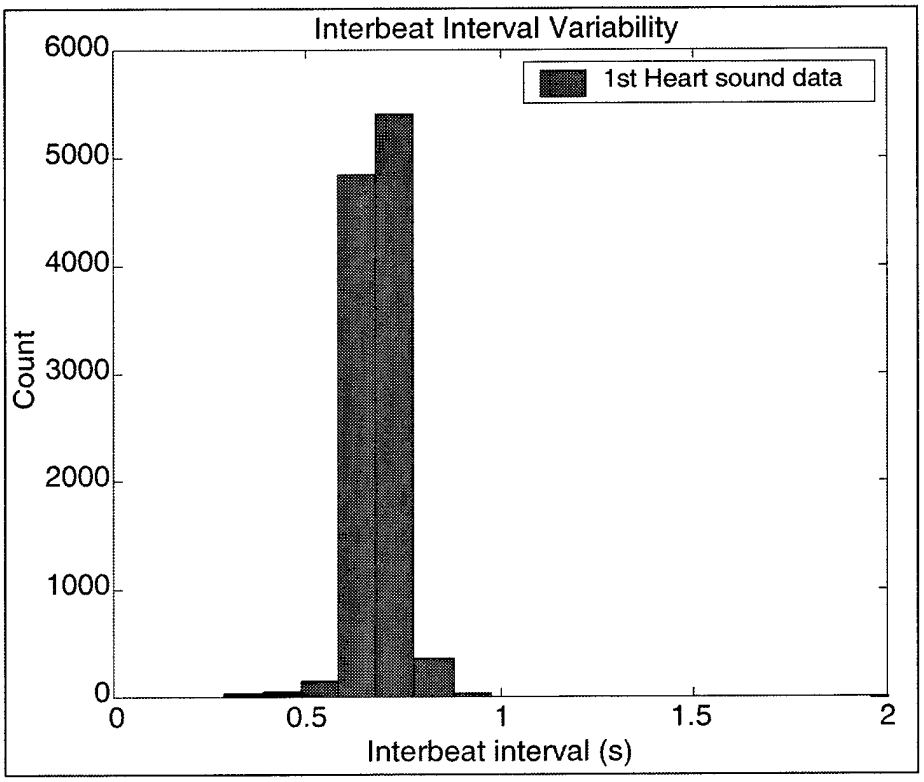

Figure 50. First heart sound variability results for experimental data taken from subject no. 2.

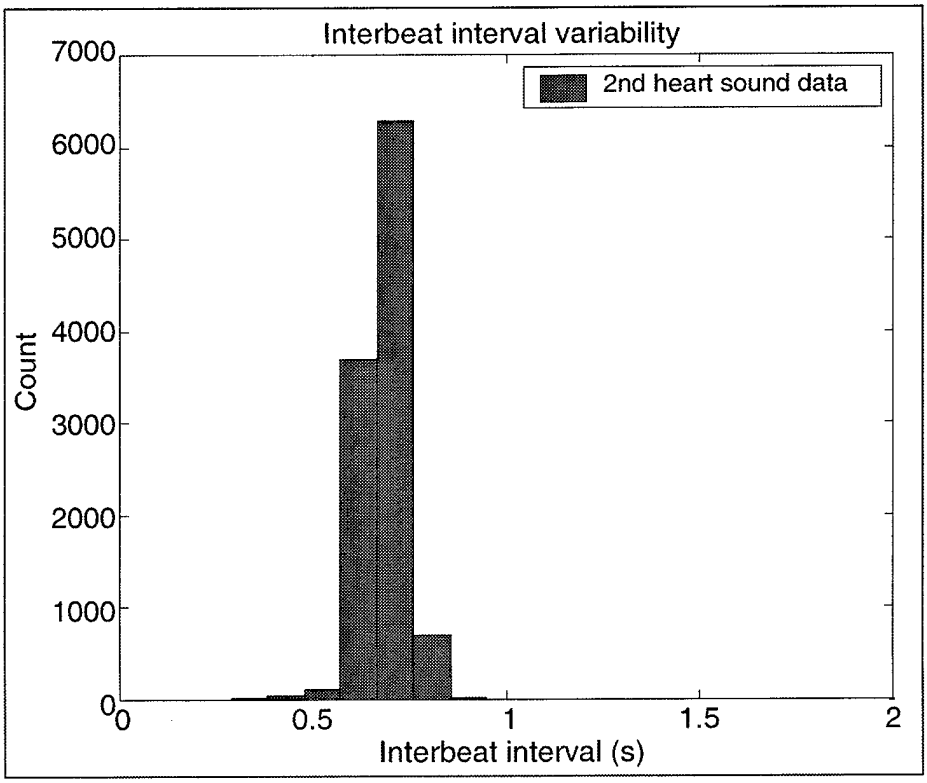

Figure 51. Second heart sound variability results for experimental data taken from subject no. 2.

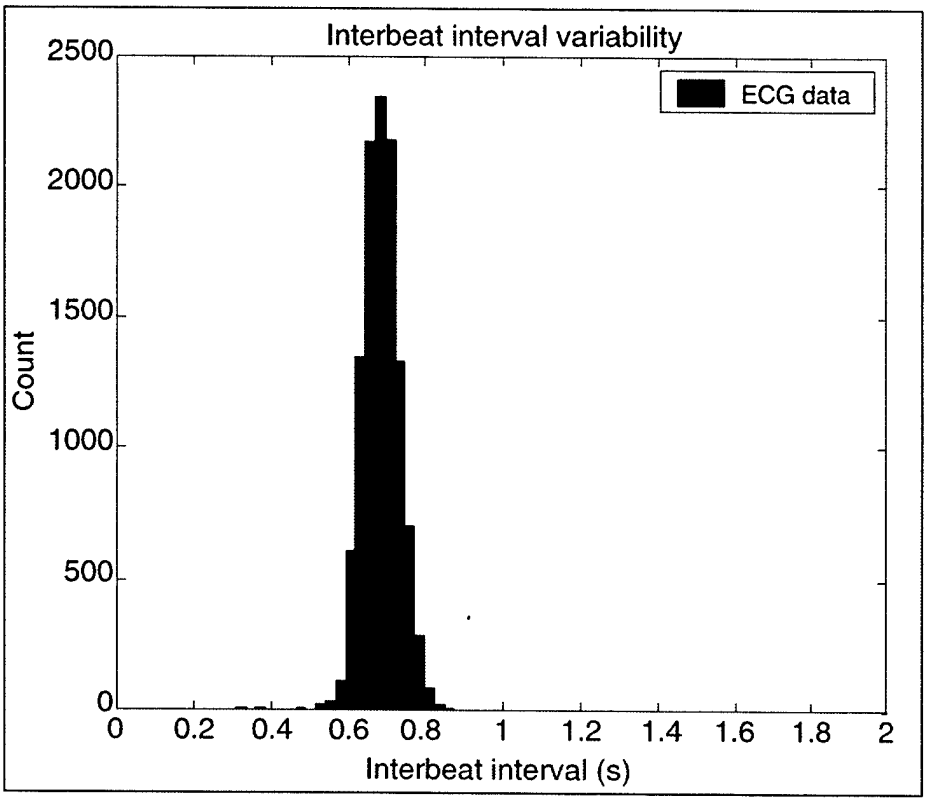

Figure 52. ECG IBI variability results for experimental data taken from subject no. 2.

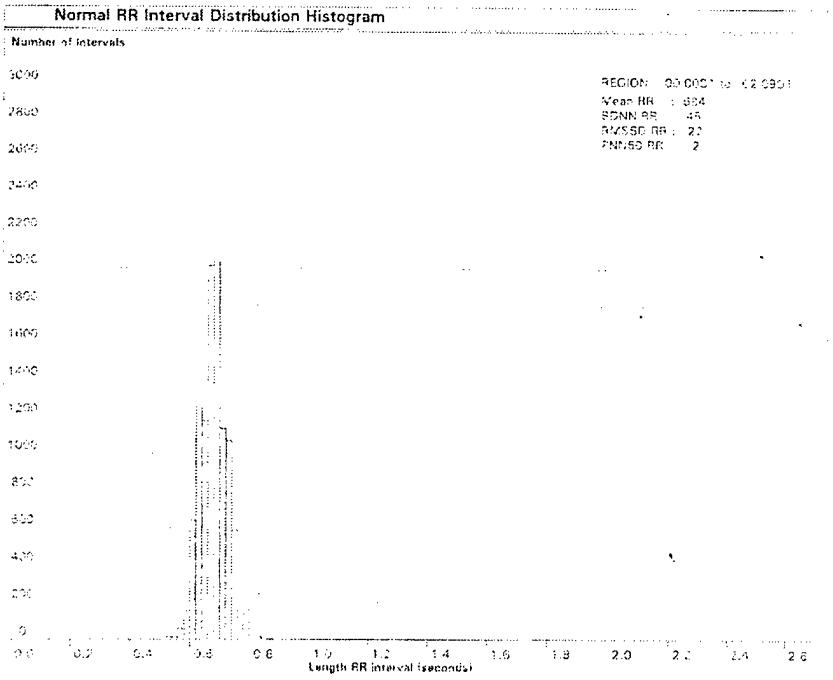

Figure 53. Holter Monitor ECG IBI variability results for experi mental data taken from subject no. 2.

Comparing the results of Figures 42 and 43 with those of Figures 44 and 45, it is observed that there is again generally good agreement between the first and second heart sound IBI extraction results and the ECG IBI heart rate results. The heart sound IBI variability appears to be about the same or slightly less for this data set from subject no. 2 than for the data set from subject no. 1. One significant difference between this data set (subject no. 2) and the previous data set (subject no. 1) is in the outliers found in the calculated ECG IBIs for this data set (subject no. 2). An examination of the prefiltered and 10- to 50-Hz band-pass filtered ECG data (subject no. 2) revealed that there was significant in-band noise (presumed to be due to a marginal ECG probe connection) which was not completely eliminated during the 10- to 50-Hz band-pass filtering process. In order to reduce the mostly transient in-band noise and to improve the overall SNR, the ECG data set from subject no. 2 was filtered with a 10- to 20-Hz band-pass filter of the same design as that shown in Figures 17 and 19. Even with this more constrained filtering, some noise transients still produced some aberrant ECG IBI data points.

One of the subgoals of the movie-watching experiment was to try to detect if a particular visual or audible stimulus caused a noticeable change in heart rate in both of the tested subjects. Referring to Figures 54 and 55, it is interesting to note that near the approximately 700- and 1400-s time frames, the heart rate of both subjects increased for a significant period of time.

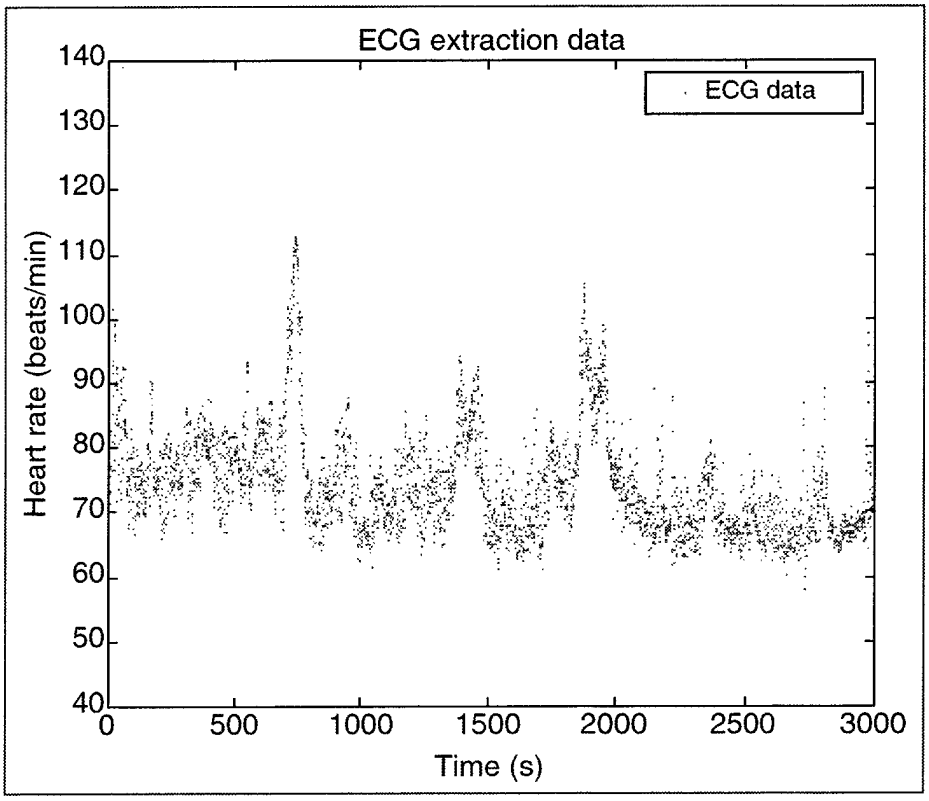

Figure 54. ECG IBI extraction results for experimental data taken from subject no. 1.

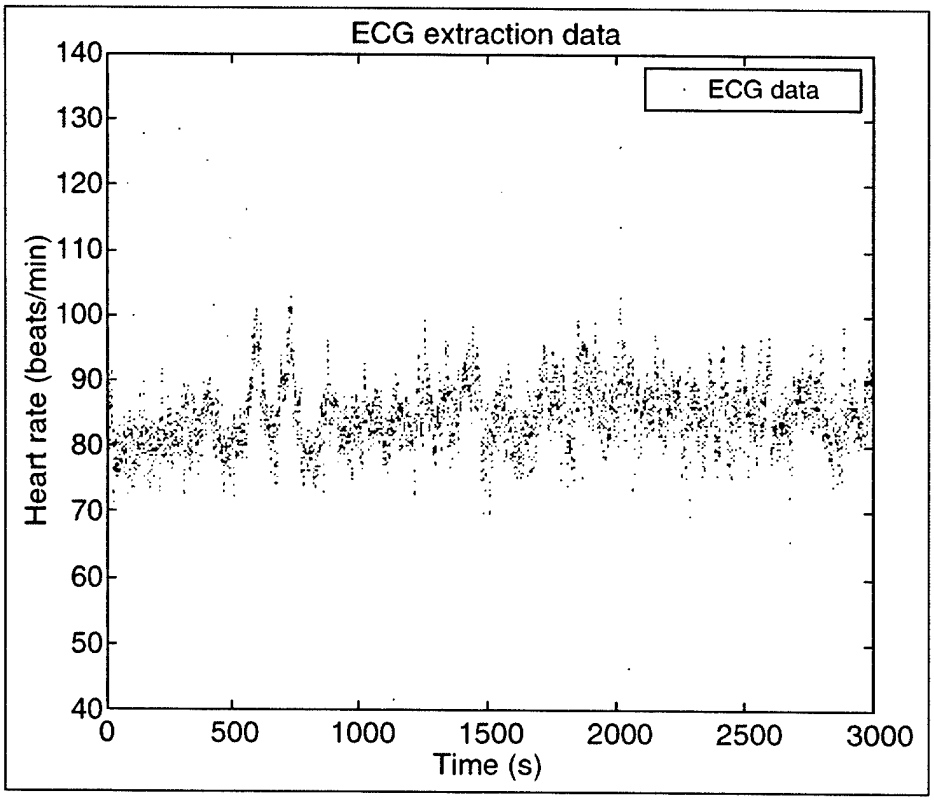

Figure 55. ECG IBI extraction results for experimental data taken from subject no. 2.

An empirical analysis of the computational breakdown of the overall algorithm yielded the following results:

- time required to execute the ECG and the acoustic band-pass filtering: approximately 6% of the total,
- time required to execute the ECG time-domain extraction process: approximately 3% of total,
- time required to execute the acoustic time-domain extraction process: approximately 9% of total, and
- time required to execute the RMS power-shaping process: approximately 82% of total.

An analysis of the RMS power-shaping process reveals that, to first order, if the RMS window is moved by only one sample between each shift, then each output sample (result) requires n^2 multiplications and n additions, where n is the length of the RMS window. Thus, if the length of the RMS window is reduced by a factor of 2, then the computational requirement would reduce by approximately a factor of 4. Attempting to reduce the computational requirements of the RMS power-shaping process in this manner would be, for the most part, counterproductive since the length of the RMS window is a somewhat critical element in the effectiveness of the shaping process. A more fruitful approach to reducing the computational requirement would be to increase the number of samples moved between each shift of the RMS window. Since the computational requirement of the process is essentially n^2 multiplications/m, where m is the number of samples moved between each RMS window shift, the computational requirement of the process can be proportionally reduced by a factor of m. The direct trade-off for increasing m is a

proportional decrease in the basic accuracy of the IBI calculations. Alternatively, the input sample stream could simply be decimated by a factor of m to yield essentially the same result.

Using a RMS power-shaping window step of two samples and a 350-MHz Pentium II-based PC, the overall Matlab-based algorithm runs at a speed of just under two times real-time in the Matlab *interpretive* environment. The algorithm is expected to run considerably faster in a compiled executable form.

8. Future Work

The movie-based data collected as a part of this project served to provide a good test bed for the baseline algorithm. The results of processing these data indicate that as long as the SNR in the in-band spectral region is greater than roughly 10 dB, this algorithm does a reasonable job of accurately extracting *instantaneous* heart sound IBI information. When a sufficient amount of in-band noise energy occurs more than once within the timeframe of the IBI extraction "region of interest," the algorithm fails.

To more reliably handle acoustic signals with lower in-band SNR, several possibilities have been identified. For example, it would be interesting to add additional criteria and logic to the present algorithms, along with some tuning parameter trials to see what amount of additional robustness could be achieved. Another exercise would be to incorporate some form of generic template or other auto- or cross-correlation process to see if the IBI information could be extracted where in-band SNR's are lower than zero dB. Another potentially useful idea would be to utilize a generic or composite "matched" filtering process prior to an auto- or cross-correlation process. Yet another potentially useful approach would be to trade-off *instantaneous* IBI tracking resolution for additional robustness by incorporating a process that uses two or more IBI periods to increase the effective SNR, if such an *averaging* IBI extraction algorithm would still meet the physiological assessment/parameter extraction goals.

In order to gain a preliminary sense of the potential benefit of incorporating an auto- or cross-correlation process into the IBI extraction algorithms, we first performed a cross-correlation process to band-pass-filtered ECG and acoustic data from the "movie" trails previously cited and then used a straightforward peak detection routine to directly extract IBI's using no identification logic or IBI criteria.* Figure 56 shows a segment of 10- to 15-Hz band-pass-filtered ECG data taken from subject no. 2. Figure 57 shows the segment of the filtered data shown in Figure 56 (starting data point: 53400) that was used as the cross-correlation template. Figures 58 and 59 show the results of the directly extracted IBIs.

Figure 60 shows a segment of 25- to 30-Hz band-pass-filtered acoustic data taken from subject no. 2. Figure 61 shows the segment of the filtered data shown in Figure 60 (starting data point: 53400) that was used as the cross-correlation template. Figures 62 and 63 show the results of the directly extracted IBIs.

*All cross-correlation algorithm development and plots were performed with the use of Labview, version 5.

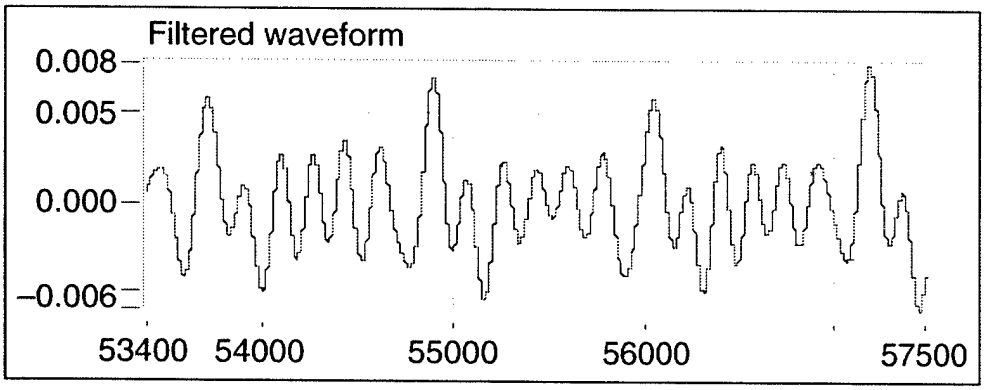

Figure 56. 10- to 15-Hz band-pass-filtered ECG waveform taken from subject no. 2.

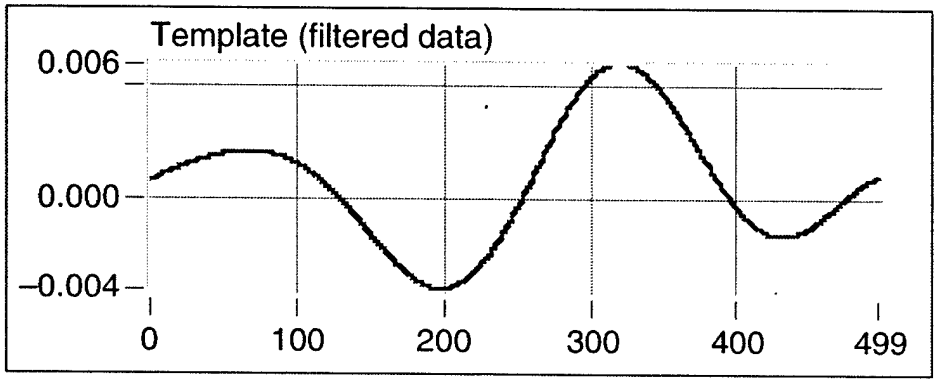

Figure 57. Segment of the filtered data shown in Figure 55 (starting data point: 53400) that was used as the cross-correlation template.

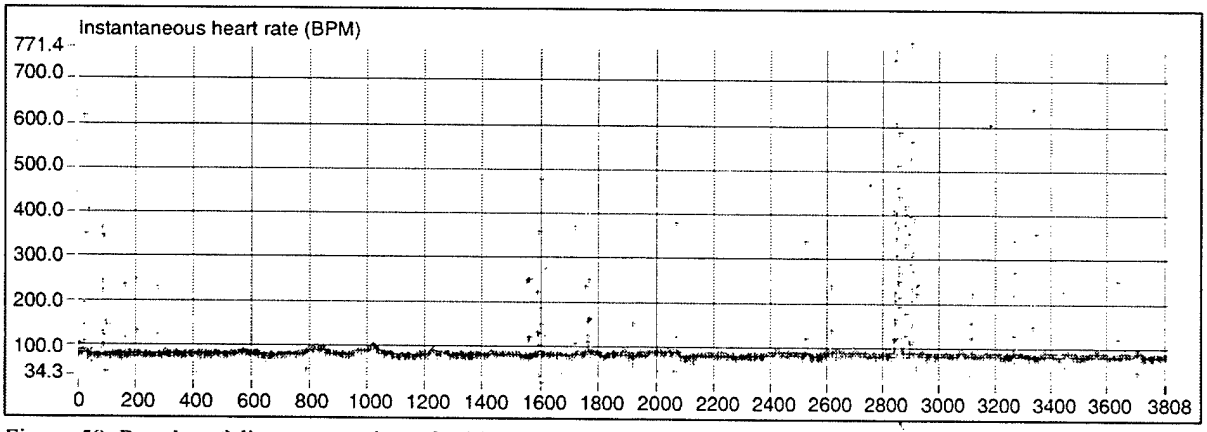

Figure 58. Results of direct extraction of subject no. 2 ECG IBIs after the cross-correlation process.

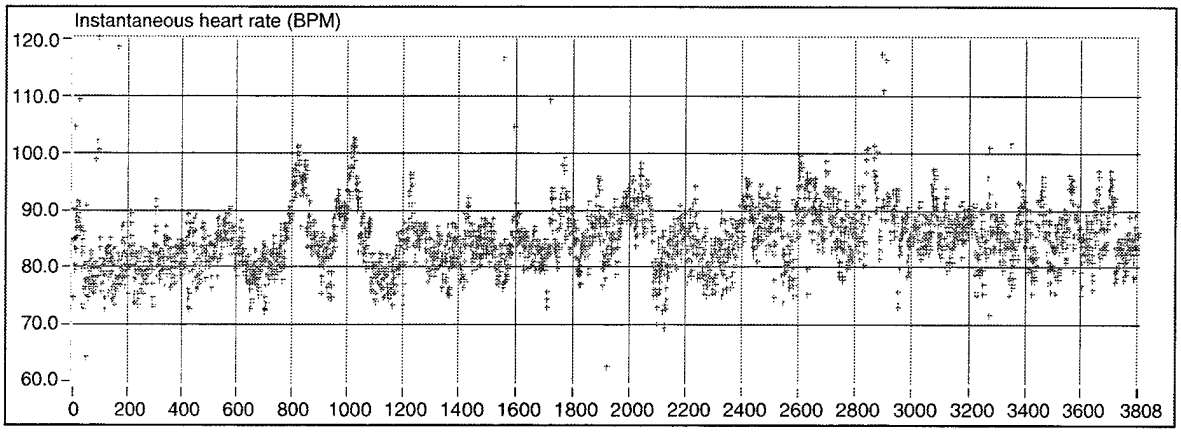

Figure 59. Results of direct extraction of subject no. 2 ECG IBIs after the cross-correlation process.

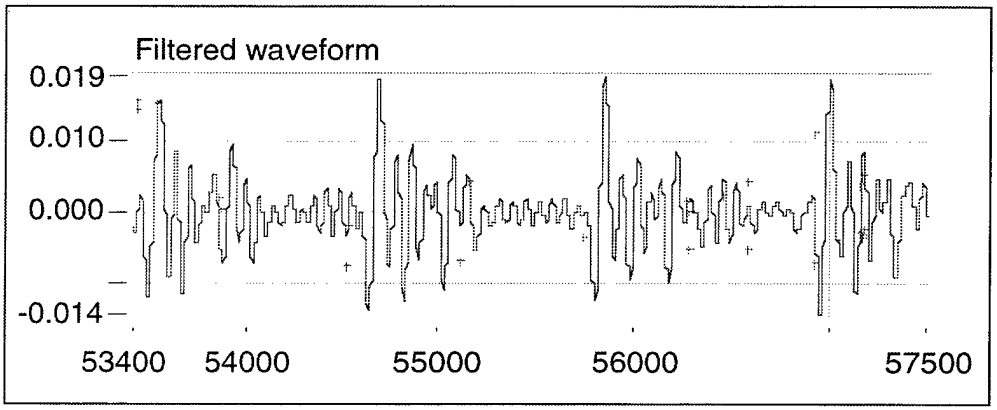

Figure 60. 25- to 30-Hz band-pass-filtered acoustic waveform taken from subject no. 2.

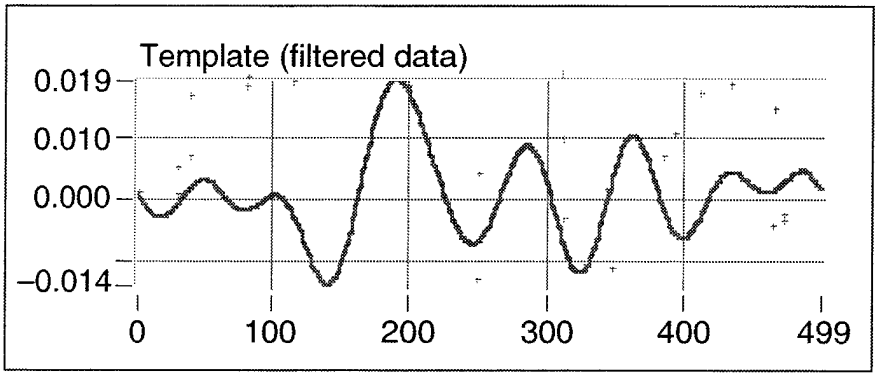

Figure 61. Segment of the filtered data shown in Figure 59 (starting data point: 54500) that was used as the cross-correlation template.

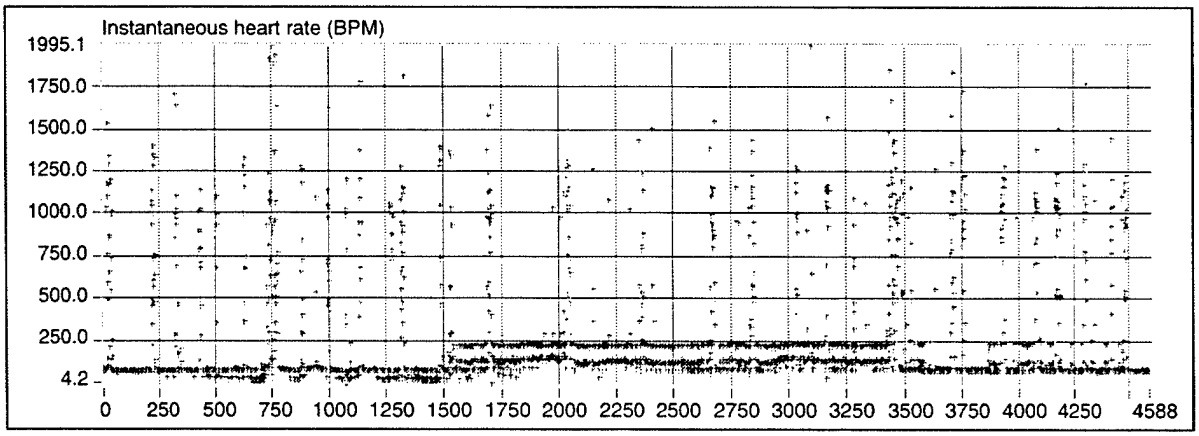

Figure 62. Results of direct extraction of subject no. 2 heart sound IBIs after the cross-correlation process.

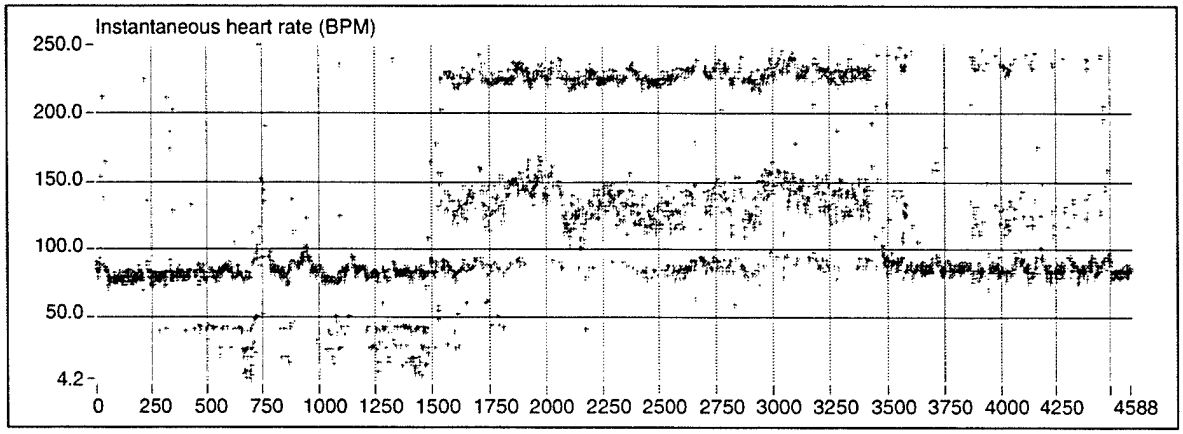

Figure 63. Results of direct extraction of subject no. 2 heart sound IBIs after the cross-correlation process.

The results shown in Figures 58, 59, 62, and 63 give good indication that a cross-correlation process, when combined with perhaps a more appropriate "matched" filtering process along with robust identification logic and IBI criteria, could perform ECG and acoustic IBI extractions *better* than the present algorithms. Examination of the results of the first cross-correlation trial which used 10- to 15-Hz band-pass-filtered ECG data, reveals that fairly accurate, fairly anomaly-free IBI extraction results were obtained without the use of *any* IBI identification logic or IBI criteria (see Figures 58 and 59). The addition of IBI identification logic and/or IBI validation criteria to the IBI extraction process used in this first cross-correlation trial could serve to eliminate a significant percentage of the relatively few anomalous IBI extractions yielding superior IBI extraction results. Examination of the results of the second cross-correlation trial using 25- to 30-Hz band-pass-filtered acoustic heart sound data, similarly reveals that fairly accurate, fairly anomaly-free, IBI extraction results would be obtained with the addition of both IBI identification logic and IBI validation criteria to the IBI extraction process (see Figures 62 and 63). This can be understood from the observation that both of the expected heart sound IBIs and all of the expected heart sound subintervals are present in the IBI/instantaneous heart rate extraction results (again, see Figures 62 and 63). For example, the interval between the first heart sound of one beat and the first heart sound of the next beat (IBI [1-1*] in Figure 1) and the interval between the second heart sound of one beat and the second heart sound of the next beat (IBI [2-2*] in Figure 1) appear in the approximately 80 beats/min regime. Additionally, the

interval between the first and second heart sound of one heart beat (IBI [1-2] in Figure 1) appears in the approximately 240 beats/min regime (as expected), while the interval between the second heart sound of one heart beat and the first heart sound of the next beat (IBI [2-1*] in Figure 1), appears in the approximately 140 beats/min regime (again, as expected). Incorporation of IBI identification logic would serve to completely eliminate the subintervals from the extraction results shown in Figures 62 and 63 while differentiating between the two basic IBIs. Incorporation of IBI validation criteria could also serve to eliminate some of the other anomalous IBI extraction results (again, shown in Figures 62 and 63).

9. Conclusions

The heart sound *instantaneous* IBI extraction algorithm developed as part of this project appears to yield good performance in environments where the in-band SNR is greater than roughly 10 dB and no more than one high-energy in-band noise burst occurs within the timeframe of the IBI extraction "region of interest." The algorithm was also shown to be capable of effectively rejecting fairly intense breath sound events. Although no deliberate attempt was made to introduce voice signals into the acoustic environment, most of the energy content of the human voice (above approximately 80 Hz) would be eliminated by, for example, the 20- to 50-Hz linear-phase band-pass filter utilized in this algorithm.

The utilization of a cross-correlation process along with perhaps a more appropriate "matched" filtering process and more robust IBI identification logic and IBI validation criteria appears to be a logical and promising progression of the present work.

10. References

1. Scanlon, M. V. "Sudden Infant Death Syndrome (SIDS) Monitor and Stimulator." U.S. Patent 5,515,865, May 1996.
2. Scanlon, M. V. "Motion and Sound Monitor and Stimulator." U.S. Patent 5,684,460, 4 November 1997.
3. Scanlon, M. V. "Acoustic Monitoring Sensor." U.S. and international patents, filed March 1996.
4. Scanlon, M. V. "Acoustic Monitoring Pad for Combat Casualty Care." Army Science Conference proceedings, Norfolk, VA, June 1996.
5. Mulder, G., and L. J. M. Mulder. "Information Processing and Cardiovascular Control." *Psychophysiology*, vol. 18, pp. 392–405, 1981.
6. Hansen, P. B., Luisada, A. A., Miltich, D. J., and Albrect, R. F. "Phonocardiography as a Monitor of Cardiac Performance During Anesthesia." *Anesthesia Analgelsia*, vol. 68, pp. 385–387, 1989.

7. Bartels, A., and D. Harder. "Non-Invasive Determination of Systolic Blood Pressure by Heart Sound Pattern Analysis." *Clinical Phys. Physiol. Meas.*, vol. 13 no. 3, pp. 249–256 1992.

8. Oppenheim, A., R. Schafer, and J. Buck. *Discrete-Time Signal Processing*. Second edition, Upper Saddle River, NJ: Prentice Hall, 1999.

Appendix A: Heart Sound IBI and ECG Heart Beat Extraction Algorithm

```
%
% Program to read in and extract parameters from ECG & Acoustic Sensor data:
%
%
% Program to design an optimal linear-phase FIR pass-band filter
% using the Parks-McClellan algorithm for the ECG data.
%
% First, estimate filter order, freq. band edges, freq. band amplitudes, and
% weights for Parks-McClellan algorithm.
%
Fs=1500                                  % Sampling frequency
Fls=0 % Low Stopband frequency edge
Flp=10                                   % Low Passband frequency edge
Fhp=50                                   % High Passband frequency edge
Fhs=60                                   % High Stopband frequency edge
A=[0,1,0]                                % Desired amplitudes
Rls=0.01                                 % Maximum low-frequency stopband
      % ripple
Rp=0.01                                  % Maximum passband ripple
Rhs=0.01                                 % Maximum high-frequency stopband
      % amplitude (ripple)
F=[Fls,Flp,Fhp,Fhs];                     % Cutoff frequencies
Dev=[Rls,Rp,Rhs];                        % Deviation vector
[n,f0,a0,w]=remezord(F,A,Dev,Fs)         % Estimate parameters
%
n_ECG=2*ceil(n/2)                        % Make n even for Type I FIR filter
%
ECG_Filter=remez(n_ECG,f0,a0,w);         % Calculate filter coefficients
%
%
% Program to design an optimal linear-phase FIR pass-band filter
% using the Parks-McClellan algorithm for the Heart Sound data.
%
% First, estimate filter order, freq. band edges, freq. band amplitudes, and
% weights for Parks-McClellan algorithm.
%
Fs=1500                                  % Sampling frequency
Fls=10                                   % Low Stopband frequency edge
Flp=20                                   % Low Passband frequency edge
Fhp=50                                   % High Passband frequency edge
Fhs=60                                   % High Stopband frequency edge
A=[0,1,0]                                % Desired amplitudes
Rls=0.01                                 % Maximum low-frequency stopband
      % ripple
Rp=0.01                                  % Maximum passband ripple
Rhs=0.01                                 % Maximum high-frequency stopband
      %amplitude (ripple)
F=[Fls,Flp,Fhp,Fhs];                     % Cutoff frequencies
Dev=[Rls,Rp,Rhs];                        % Deviation vector
[n,f0,a0,w]=remezord(F,A,Dev,Fs)         % Estimate parameters
%
n_Acu=2*ceil(n/2)                        % Make n even for Type I FIR filter
%
Acu_Filter=remez(n_Acu,f0,a0,w);         % Calculate filter coefficients
```

```
%
%
% Program to read in and plot ECG & Acoustic Data
%
readfile=input('Enter filename (w/ extension) to read (use single quotes): ')
%
%filename='sgl54k.bin'
%
writefile1='ECG.bin'                          %
writefile2='Acu.bin'                          %
fid1=fopen(readfile)                          % Get read file ID.
[readfile,permission,architecture]=fopen(fid1)  % Get permission, architecture.
permission='rb'                               % Set permission for read binary
architecture='ieee-be'                        % Set architecture to IEEE Big
      % Endian format.
[fid1,message]=fopen(readfile,permission,architecture)   % Define architecture, fid of read
      % file.
permission='wb'                               % Set permission for write binary
architecture='native'                         % Set architecture for local
      % machine
[fid2,message]=fopen(writefile1,permission,architecture)  % Define architecture, fid of
      % write file.
[fid3,message]=fopen(writefile2,permission,architecture)  % Define architecture, fid of
      % write file.
precision='float32'
%
[A,count]=fread(fid1,900000,precision);       % Read data block into array A
      % using 32-bit
      % floating-point precision.
%
Acoustic1=[];                                 % Define, zero main arrays.
ECG1=[];                                      %
Acu_Carryover=[];                             % Acoustic carryover data from
      % end of last buffer
ECG_Carryover=[];                             % ECG carryover data from end
      % of last buffer
ECG_Time_Offset=0;                            %
HS_Time_Offset=0;                             %
%
Fs=1500;                                      % Sample frequency (Hz)
%
while count > 0                               %
  Acoustic1=cat(1,Acu_Carryover,A(2:2:count));  % Concatenate carryover buffers
      % to front of
  ECG1=cat(1,ECG_Carryover,A(1:2:count));     % most-recently-read acoustic & ECG data blocks.
  %
  % Process ECG data
  %
  Sum1=sum(ECG1);                             % Remove DC offset
  DC_Offset=Sum1/length(ECG1);                % from ECG data.
  %
  ECG2=(-1)*(ECG1-DC_Offset);                 %** Key Parameter **
  %
  ECG_New=filter(ECG_Filter,1,ECG2);          % Filter ECG data.
  ECG_New=cat(1,ECG_New,(zeros(1,n_ECG/2))');  % Zero-pad filtered ECG data;
  ECG_New=ECG_New(n_ECG/2+1:1:end);           % Subtract Group delay
  %
  ECG_New=(ECG_New+abs(ECG_New))./2;          % Clip filtered ECG data
  %
```

```
%
% Beginning of ECG Rn to Rn+1 extraction algorithm
%
% Initialize variables
%
ECG=[];                                          %
Start_Window=1;                                  % Pointer to start of extraction window
Max_length=fix((1/20)*60*4*Fs);                  % Initialize to capture a minimum
    % of 3 beats occuring at lowest
    % heart rate of 20 beats/min.
    % = 1/(20 beats/min)*60 sec/min*4 beats*sample rate
Ext_Window_length=Max_length;                    %
Last_ECG=1;                                      %
%
QRS_Width=100;                                   % ** Key Tuning Parameter **
                                                 % Set peak detection blanking half width
                                                 % (Can set up to ~ 200)
%
Detect=0;                                        % Set last heart beat detection flag = false
%
while Start_Window+Ext_Window_length < length(ECG_New);
  %
  % Determine amplitude window in which to locate the peak responses of ECG waveform beats.
  %
  Ext_Window=ECG_New(Start_Window:1:Start_Window+Ext_Window_length-1);
  Max_Amp=max(Ext_Window);                       % Set upper peak detection threshold
  DC_Avg=sum(Ext_Window)/length(Ext_Window);
                                                 %
  Min_Amp=3*DC_Avg+0.3*(Max_Amp-DC_Avg);         % ** Key Tuning Parameter **
                                                 % Set lower peak detection threshold
  Top=[];
  Bottom=[];
  Top(1:length(Ext_Window))=Max_Amp;
  Bottom(1:length(Ext_Window))=Min_Amp;
  %
  % Locate all ECG peaks within extraction window.
  %
  Peaks=[];
  [Peak,Index]=max(Ext_Window);
  while Peak > Min_Amp
    Peaks=cat(2,Peaks,Index+Start_Window);
    Min_zero=Index-QRS_Width;
    if Min_zero < 1
      Min_zero=1;
    end
    Max_zero=Index+QRS_Width;
    if Max_zero > Ext_Window_length;
      Max_zero=Ext_Window_length;
    end
    Ext_Window(Min_zero:1:Max_zero)=0;
    [Peak,Index]=max(Ext_Window);
    %
  end
  %
  % Evaluate temporal locations of first two heart beats within
  % the window to determine if they meet criteria
  %
```

```
  if length(Peaks) < 3                                    % If extraction window is
                                                          % too short.
    if Ext_Window_length >= Max_length                    % increase extraction window length.
      Start_Window=ceil(Start_Window+Max_length/3);       % advance (next) start point.
    else
      Ext_Window_length=ceil(Ext_Window_length*3/(length(Peaks)+0.01));
      if Ext_Window_length >= Max_length
        Ext_Window_length=fix(Max_length);
      end
    end
  else
    %
    Peaks=sort(Peaks);                                    % Sort time location of
                                                          %  peaks
                                                          % in ascending order.
    if Detect==1;
      New_ECG(1)=Last_ECG;                %
      New_ECG(2)=Peaks(1);                %
    else
      New_ECG(1:2)=Peaks(1:2);            %
    end
    %
    P1_P2=New_ECG(2)-New_ECG(1);          % Criteria:
                                          % P1_P2 must be < 20 beats/min interval and
                                          % P1_P2 must be > 240 beats/min interval.
    %
    if P1_P2 < (1/20)*60*Fs & P1_P2 > (1/240)*60*Fs
      % Criteria has been met!
      Last_ECG=New_ECG(2);
      Detect=1;                           % Set Valid detection for this pass.
                                          %
      Start_Window=ceil(Last_ECG+QRS_Width);  % ** Key Parameter **

                                          %
      Ext_Window_length=ceil((New_ECG(2)-New_ECG(1))*4);  % Adjust window length.
      New_ECG=New_ECG./Fs;                % Scale to units of seconds.
      New_ECG=New_ECG+ECG_Time_Offset;    % Adjust to global time.
      ECG=cat(2,ECG,New_ECG(1),New_ECG(2));  % Store times of valid ECG pairs.
    else
      Start_Window=ceil(New_ECG(1)+QRS_Width);
      Detect=0;                           % Set valid detection flag for
                                          % this pass = false.
    end
    %
  end
  %
  New_ECG(1),New_ECG(2)
  %pause
end
%
ECG_Carryover=ECG1(Start_Window:1:length(ECG1));  % Form carryover buffer
ECG_Time_Offset=ECG_Time_Offset+Start_Window/Fs;  % Set ECG global time offset
ECG1=[];                                          %
%
% Process Acoustic Data
%
Sum2=sum(Acoustic1);                              % Remove DC offset
```

```
DC_Offset=Sum2/length(Acoustic1);                    % from acoustic data
Acoustic2=Acoustic1-DC_Offset;
%
Acoustic_New=filter(Acu_Filter,1,Acoustic2);         % Filter Acoustic data
Acoustic_New=cat(1,Acoustic_New,(zeros(1,n_Acu/2))'); % Zero-pad filtered ECG data;
Acoustic_New=Acoustic_New(n_Acu/2+1:1:end);          % Subtract Group delay
%
% Apply an RMS power-shaping algorithm to the filtered acoustic data to
% find the centroids of the Heart Sound burst energy.
%
RMS_Window_length=60;                                % ** Key Shaping Parameters **
RMS_Window_overlap=58;
I1=1;
Acoustic_New2=zeros(1,length(Acoustic_New));
if   RMS_Window_length<length(Acoustic_New);
  Acu_temp=Acoustic_New(1:1:RMS_Window_length);
  RMS=sqrt(mean(Acu_temp.*Acu_temp));                % Calculate the RMS power
  Acoustic_New2(1:1:RMS_Window_length)=RMS;          % within the RMS window.
  I1=I1+RMS_Window_length-RMS_Window_overlap;        % Set first RMS window length values
                                                     % of new acoustic array equal to the
                                                     % RMS power within the RMS window.

else
  error('Window length longer than signal');
end
while I1+RMS_Window_length-1<length(Acoustic_New)
  Acu_temp1=Acoustic_New(I1:1:I1+RMS_Window_length-1);
  RMS=sqrt(mean(Acu_temp1.*Acu_temp1));
  Acu_temp2(1:1:RMS_Window_length)=RMS;
  Acoustic_New2(I1:1:I1+RMS_Window_length-1)=Acoustic_New2(I1:1:I1+RMS_Window_length-
1)+Acu_temp2(1:1:RMS_Window_length);

                                                     % Add new RMS power values to the new acoustic
                                                     % array after shifting by difference between
                                                     % the RMS_Window_length and the
RMS_Window_overlap.
  I1=I1+RMS_Window_length-RMS_Window_overlap;
end
%
%
% Beginning of Heart Sound extraction algorithm
%
% Initialize variables
%
HS1=[];

                                                     % First heart sound time vector
HS2=[];                                              % Second heart sound time vector
Start_Window=1;                                      % Pointer to start of extraction window
Max_length=fix((1/20)*60*4*Fs);                      % Initialize to capture a minimum
                                                     % of 3 heart sound pairs occuring at
                                                     % lowest heart rate of 20 beats/min.
                                                     % = 1/(20 beats/min)*60 sec/min*4 beats*sample
                                                     % rate
```

```
Ext_Window_length=Max_length;                                   %
Detect=0;                                                       % First heart sound detection flag = false
%
while Start_Window+Ext_Window_length < length(Acoustic_New2);
  %
  % Determine amplitude window in which to locate the peak responses of heart sounds.
  %
  Ext_Window=Acoustic_New2(Start_Window:1:Start_Window+Ext_Window_length-1);
  Max_Amp=max(Ext_Window);                                      % Set upper peak detection threshold
  DC_Avg=sum(Ext_Window)/length(Ext_Window)                     %
  Min_Amp=1.2*DC_Avg + 0.0*(Max_Amp-DC_Avg);                    % ** Key Tuning Parameter **
                                                                % Set lower peak detection threshold
                                                                %
  Top=[];                                                       % Form Max. and Min. threshold
  Bottom=[];                                                    % bars for display.
  Top(1:length(Ext_Window))=Max_Amp;
  Bottom(1:length(Ext_Window))=Min_Amp;
  %
  %
  % Locate all heart sound peaks within extraction window.
  %
  Peaks=[];
  [Peak,Index]=max(Ext_Window);                                 % Locate the peak of the RMS Heart Sound pulse
  while Peak > Min_Amp                                          % with the largest value above the Min. threshold.
    Peaks=cat(2,Peaks,Index+Start_Window);
                                                                %
    Min_zero=fix(Index-3.5*RMS_Window_length);                  % ** Key Tuning Parameter **
                                                                %
    if Min_zero < 1
      Min_zero=1;
    end
    Max_zero=fix(Index+3.5*RMS_Window_length);
    if Max_zero > Ext_Window_length;
      Max_zero=fix(Ext_Window_length);
    end
    Ext_Window(Min_zero:1:Max_zero)=0;                          % Remove RMS Heart Sound pulse from the set.
    [Peak,Index]=max(Ext_Window);                               % Locate the next-largest peak above threshold.
    %
  end
  %
  % Evaluate temporal locations of first four hearts sounds to
  % determine if they meet criteria
  %
  if length(Peaks) < 7
    if Ext_Window_length >= Max_length
      Start_Window=fix(Start_Window+Max_length/7);              % Advance start point of extraction window.
    else
      Ext_Window_length=ceil(Ext_Window_length*7/(length(Peaks)+0.01));     % Increase extraction window
                                                                            % length.
```

```
      if Ext_Window_length >= Max_length
        Ext_Window_length=fix(Max_length);
      end
    end
  else
    %
    Peaks=sort(Peaks);                                  % Sort time location of peaks
                                                        % in ascending order.
    if Detect==1;                                       % Heart Sound pairs previously detected?
      New_HS(1)=Last_HS3;                               % Align first & second heart sounds detected

      New_HS(2)=Last_HS4;                               % in this pass to third and fourth heart sounds
      New_HS(3:7)=Peaks(1:5);                           % detected in previous pass.
    else
      New_HS(1:7)=Peaks(1:7);                           %
    end
    %
    P1_P2=New_HS(2)-New_HS(1);                          % Criteria:
    P2_P3=New_HS(3)-New_HS(2);                          % P1_P2 must be < P2_P3 and
    P3_P4=New_HS(4)-New_HS(3);                          % P3_P4 must be < P2_P3 and
    P4_P5=New_HS(5)-New_HS(4);                          % P3_P4 must be < P4_P5 and
    P5_P6=New_HS(6)-New_HS(5);                          % P5_P6 must be < P4_P5.
    P1_P3=New_HS(3)-New_HS(1);                          %
    P2_P4=New_HS(4)-New_HS(2);                          %
    I_max=(1/20)*60*Fs;                                 % Maximum interval between beats (@ 20 beats/
                                                        % min.)
    I_min=(1/240)*60*Fs;                                % Minimum interval between beats (@ 240 beat/
                                                        % min.).
    %
    if (P1_P2<P2_P3 & P3_P4<P2_P3 & P3_P4<P4_P5 & P5_P6<P4_P5) & (P1_P3<I_max & P1_P3>I_min &
P2_P4<I_max & P2_P4>I_min)
      % Criteria has been met!
      Last_HS3=New_HS(3);
      Last_HS4=New_HS(4);
      Detect=1;                                         % Set Valid detection for this pass.
                                                        %
      Start_Window=fix(Last_HS4+1.4*RMS_Window_length); % ** Key Parameters **
      Ext_Window_length=fix((New_HS(3)-New_HS(1))*5);   % Adjust window length.
                                                        %
      New_HS=New_HS./Fs;                                % Scale to units of seconds.
      New_HS=New_HS+HS_Time_Offset;                     % Adjust to global time.
      HS1=cat(2,HS1,New_HS(1),New_HS(3));               % Store times of valid HS pairs.
      HS2=cat(2,HS2,New_HS(2),New_HS(4));
    else
      %if length(Peaks)<7       %
      %if Ext_Window_length>= Max_length
      %       Start_Window=ceil(New_HS(1)+1.4*RMS_Window_length);
      %else
      %      Ext_Window_length=Ext_Window_length*7/length(Peaks);
      %       if Ext_Window_length >= Max_length
      %             Ext_Window_length=fix(Max_length);
      %      end
      %end
      %else
      %
      if Detect~=1
        % Check to see if first 4 HS's are out of phase by one sound.
        Start_Window=fix(New_HS(1)+1.4*RMS_Window_length);
      else
```

```
        New_HS(3:6)=New_HS(4:7);                          % Check for extra peak (noise)
                                                          % between Peaks 2 & 3.
                                                          %
        P1_P2=New_HS(2)-New_HS(1);      % Criteria:
        P2_P3=New_HS(3)-New_HS(2);      % P1_P2 must be < P2_P3 and
        P3_P4=New_HS(4)-New_HS(3);      % P3_P4 must be < P2_P3 and
        P4_P5=New_HS(5)-New_HS(4);      % P3_P4 must be < P4_P5 and
        P5_P6=New_HS(6)-New_HS(5);      % P5_P6 must be < P4_P5.
        P1_P3=New_HS(3)-New_HS(1);      %
        P2_P4=New_HS(4)-New_HS(2);      %
        I_max=(1/20)*60*Fs;             % Maximum interval between beats (@ 20 beats/min.)
        I_min=(1/240)*60*Fs;            % Minimum interval between beats (@ 240 beat/ min.).
        %
        if (P1_P2<P2_P3 & P3_P4<P2_P3 & P3_P4<P4_P5 & P5_P6<P4_P5) & (P1_P3<I_max & P1_P3>I_min &
P2_P4<I_max & P2_P4>I_min)
          % Criteria has been met!
          Last_HS3=New_HS(3);
          Last_HS4=New_HS(4);
          Detect=1;                                       % Valid detection on previous attempt.
          Start_Window=fix(Last_HS4+1.4*RMS_Window_length);
          New_HS=New_HS./Fs;                              % Scale to units of seconds.
          New_HS=New_HS+HS_Time_Offset;                   % Adjust to global time.
          HS1=cat(2,HS1,New_HS(1),New_HS(3));             % Store times of valid HS


   pairs.
          HS2=cat(2,HS2,New_HS(2),New_HS(4));
        else
          Detect=0;                                       % Set valid detection flag for  this pass
                                                          % = false.
          % Check to see if first 4 HS's are out of phase by one sound.
          Start_Window=fix(New_HS(1)+1.4*RMS_Window_length);
          %
        end
        %
      end
      %
    end
    %
    New_HS(1:2)
    %pause
  end
                                                          %
%
Acu_Carryover=Acoustic1(Start_Window:1:length(Acoustic1));    % Form carryover buffer
HS_Time_Offset=HS_Time_Offset+Start_Window/Fs;                % Set Acoustic global time offset
Acoustic1=[];                                                 %
%
%
fwrite(fid2,ECG,precision);                                   % Write ECG data to file
%
HS(1:2:2*length(HS1))=HS1;                                    % Merge HS1 & HS2 data
HS(2:2:2*length(HS1))=HS2(1:1:length(HS1));                   %
fwrite(fid3,HS,precision);                                    % Write Acoustic data to file

ECG=[];                                                       % Clear results vectors
HS1=[];                                                       %
HS2=[];                                                       %
```

```
HS=[];
%
[A,count]=fread(fid1,900000,'float32');

end
%
status=fclose(fid1)
status=fclose(fid2)
status=fclose(fid3)

%
writefile1='ECG.bin';
writefile2='Acu.bin';
permission='rb';
architecture='native';

[fid2,message]=fopen(writefile1,permission,architecture);
[fid3,message]=fopen(writefile2,permission,architecture);
precision='float32';
%
[ECG,count]=fread(fid2,inf,precision);

[HS,count]=fread(fid3,inf,precision);

%
status=fclose(fid2)
status=fclose(fid3)
%
HS1=HS(1:2:length(HS));
HS2=HS(2:2:length(HS));
%
%
% Plot Heart rate (Beats/min.) vs. Time (sec.) for ECG data
%
Time1=ECG(1:2:length(ECG));
Rate1=60./(ECG(2:2:length(ECG))-ECG(1:2:length(ECG)));
subplot(1,1,1);
plot(Time1,Rate1);
xlabel('Time (sec)');
ylabel('Heart Rate (beats/min)');
Title('ECG Extraction Data');
pause
%
% Plot Poincare Heart Rate Variability for ECG data
%
ECG_I1=1000.*(ECG(2:4:4*fix(length(ECG)/4))-ECG(1:4:4*fix(length(ECG)/4)));
ECG_I2=1000.*(ECG(4:4:4*fix(length(ECG)/4))-ECG(3:4:4*fix(length(ECG)/4)));
x=[0,2000];
y=[0,2000];
plot(ECG_I2,ECG_I1,'b+',x,y);
axis([0,2000,0,2000]);
xlabel('Rn to Rn+1 (msec)');
ylabel('Rn-1 to Rn (msec)');
Title('Poincare Heart Rate Variability');
pause
```

```
%

% Read data into array A using 32-bit
% floating-point precision.


% Close all files
%
%


%
%
% Set permission for read binary

% Set architecture for local machine
% Define architecture, fid of ECG file.
% Define architecture, fid of Acoustic file.
%

% Read data into array ECG using 32-bit
% floating-point precision.
% Read data into array HS using 32-bit
% floating-point precision.

% Close all files
%

% Get HS1 Acoustic data
% Get HS2 Acoustic data
```

```
%
% Plot Interbeat Interval Histogram for ECG data
%
ECG_IBI=60./Rate1;
hist(ECG_IBI);
xlabel('Interbeat Interval (sec)');
ylabel('Count');
Title('Interbeat Interval Variability');
pause
%
% **************
%
% Plot Heart rate (Beats/min.) vs. Time (sec.) for HS1 & HS2 data
%
Time1=HS1(1:2:length(HS1));
Rate1=60./(HS1(2:2:length(HS1))-HS1(1:2:length(HS1)));
Time2=HS2(1:2:length(HS2));
Rate2=60./(HS2(2:2:length(HS2))-HS2(1:2:length(HS2)));
%
subplot(1,1,1);
plot(Time1,Rate1,Time2,Rate2);
xlabel('Time (sec)');
ylabel('Heart Rate (beats/min)');
Title('Acoustic Extraction Window');
pause
%
% Plot Poincare Heart Rate Variability for HS1 & HS2 data
%
HS1_I1=1000.*(HS1(2:4:4*fix(length(HS1)/4))-HS1(1:4:4*fix(length(HS1)/4)));
HS1_I2=1000.*(HS1(4:4:4*fix(length(HS1)/4))-HS1(3:4:4*fix(length(HS1)/4)));
x=[0,2000];
y=[0,2000];
plot(HS1_I2,HS1_I1,'b+',x,y);
axis([0,2000,0,2000]);
xlabel('HSn to HSn+1 (msec)');
ylabel('HSn-1 to HSn (msec)');
Title('Poincare Heart Rate Variability');
pause
HS2_I1=1000.*(HS2(2:4:4*fix(length(HS2)/4))-HS2(1:4:4*fix(length(HS2)/4)));
HS2_I2=1000.*(HS2(4:4:4*fix(length(HS2)/4))-HS2(3:4:4*fix(length(HS2)/4)));
x=[0,2000];
y=[0,2000];
plot(HS2_I2,HS2_I1,'b+',x,y);
axis([0,2000,0,2000]);
xlabel('HSn to HSn+1 (msec)');
ylabel('HSn-1 to HSn (msec)');
Title('Poincare Heart Rate Variability');
pause
%
% Plot Interbeat Interval Histogram for HS1 & HS2 data
%
HS1_IBI=60./Rate1;
hist(HS1_IBI);
xlabel('Interbeat Interval (sec)');
ylabel('Count');
Title('Interbeat Interval Variability');
pause
HS2_IBI=60./Rate2;
hist(HS2_IBI);
xlabel('Interbeat Interval (sec)');
```

```
ylabel('Count');
Title('Interbeat Interval Variability');
pause
%
% ***********
%
```

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	October 2002	Progress, 01 Jan 2000 to 30 July 2001

4. TITLE AND SUBTITLE
Design of a Heart Sound Extraction Algorithm for an Acoustic-Based Health Monitoring System

5. FUNDING NUMBERS
DA PR: N/A
PE: 63739E

6. AUTHOR(S)
Steven R. Murrill and Michael V. Scanlon

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
Attn: AMSRL- SE-EE
Adelphi, MD 20783-1197

8. PERFORMING ORGANIZATION REPORT NUMBER
ARL-MR-517

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
U.S. Army Research Laboratory
Adelphi, MD 20783-1197

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES
ARL PR: 1N1VCC
AMS code: N/A

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited.

12b. DISTRIBUTION CODE

13. ABSTRACT *(Maximum 200 words)*

The U.S. Army Research Laboratory is developing sensor technology to monitor the soldier's physiological variables and motor activities by gathering and analyzing acoustic data. The sensor/transducer consists of a fluid or gel contained within a small, conformable, rubber bladder or pad that also includes a hydrophone. This enables the collection of high signal-to-noise-ratio (SNR) cardiac, respiratory, voice, and other physiological data. The pad minimizes interference from ambient noise because it couples poorly with airborne noise.

A heart sound interbeat interval extraction algorithm has been developed using only basic concepts and techniques generally known/available to first-semester Discrete-Time Signal Processing course graduates. The algorithm appears to yield good performance in environments where the in-band SNR is greater than roughly 10 dB and no more than one high-energy, in-band noise burst occurs within the timeframe of the IBI extraction "region of interest." The algorithm has been shown to be capable of effectively rejecting fairly intense breath sound events. Although no deliberate attempt was made to introduce voice signals into the acoustic environment, most of the energy content of the human voice (above approximately 80 Hz) would be eliminated by the 20- to 50-Hz linear-phase, band-pass filter utilized in this algorithm.

14. SUBJECT TERMS	15. NUMBER OF PAGES
acoustic, physiological, sensor, heart sound, extraction algorithm	63
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified	Unclassified	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std. Z39-18
298-102